(19)日本国特許庁（ＪＰ）　(12)**公開特許公報（Ａ）**　(11)特許出願公開番号
特開2001－154852
（P2001－154852A）
(43)公開日　平成13年６月８日(2001.6.8)

(51)Int.Cl.[7]	識別記号	ＦＩ		ﾃｰﾏｺｰﾄﾞ(参考)
Ｇ０６Ｆ　9/45		Ｃ０６Ｆ　9/44	３２０Ａ	

審査請求　有　　請求項の数91　ＯＬ　（全 85 頁）

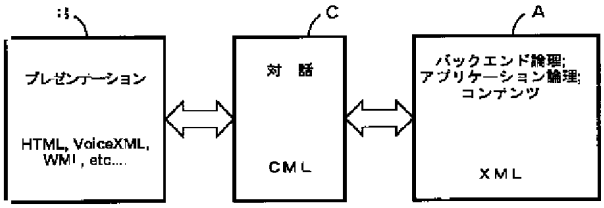
(21)出願番号　特願2000－311661(P2000－311661)
(22)出願日　平成12年10月12日(2000.10.12)
(31)優先権主張番号　６０／１５８７７７
(32)優先日　平成11年10月12日(1999.10.12)
(33)優先権主張国　米国（ＵＳ）
(31)優先権主張番号　０９／５４４８２３
(32)優先日　平成12年４月６日(2000.4.6)
(33)優先権主張国　米国（ＵＳ）

(71)出願人　390009531
インターナショナル・ビジネス・マシーンズ・コーポレ－ション
ＩＮＴＥＲＮＡＴＩＯＮＡＬ　ＢＵＳＩＮＥＳＳ　ＭＡＳＣＨＩＮＥＳ　ＣＯＲＰＯＲＡＴＩＯＮ
アメリカ合衆国10504、ニューヨーク州　アーモンク　（番地なし）
(74)代理人　100086243
弁理士　坂口　博　（外２名）

最終頁に続く

(54)【発明の名称】　マルチモーダル・ブラウジングおよび会話型マークアップ言語の実施の方法およびシステム

(57)【要約】　（修正有）
【課題】　ユーザが任意のタイプの情報にアクセスするために利用している任意の装置とのユーザ対話に基づく、新しいアプリケーション・プログラミング言語を提供すること。
【解決手段】　この新しい言語は、本明細書で「会話型マークアップ言語（ＣＭＬ）」と呼ぶ。好ましい一実施形態では、ＣＭＬは、ユーザが所与のコンピューティング装置と行うことになる「ダイアログ」または「会話」を表すためのハイレベルＸＭＬベースの言語である。アプリケーション作成者は、「会話型ジェスチャ」と呼ぶ対話ベースの要素を使用してアプリケーションをプログラムすることができる。本発明はまた、様々なモーダル性特定の表現、例えばＨＴＭＬベースのグラフィカル・ユーザ・インタフェース（ＧＵＩ）ブラウザ、VoiceXＭＬベースのスピーチ・ブラウザなどに従ってＣＭＬの特徴をサポートすることのできるマルチモーダル・ブラウザの様々な実施形態を可能にする。



【特許請求の範囲】
【請求項１】ユーザから１つまたは複数のコンピュータ・ベースの装置を介してアクセス可能なアプリケーションをプログラムする方法であって、
前記アプリケーションにアクセスするために使用される前記１つまたは複数のコンピュータ・ベースの装置とユーザが対話ベースのプログラミング・コンポーネントによって行うことが可能な対話を表現するステップを含み、
前記対話ベースのプログラミング・コンポーネントが、前記アプリケーションに関連するコンテンツ／アプリケーション論理およびプレゼンテーション要件に対して独立、さらに、コンポーネントごとにトランスコーディングされて、前記１つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの１つまたは複数のモーダル性特定のレンダリングを生成する方法。
【請求項２】前記アプリケーションの少なくとも一部がサーバから、クライアントの役割を果たす前記１つまたは複数のコンピュータ・ベースの装置のうちの少なくとも１つにダウンロードされるクライアント／サーバ構成において、前記サーバに常駐する前記コンテンツ／アプリケーション論理への接続を提供するように動作可能なコードを前記アプリケーション中に含めるステップをさらに含む、請求項１に記載の方法。
【請求項３】前記コンテンツ／アプリケーション論理接続コードが、前記アプリケーションに関連する１つまたは複数のデータ・モデル、属性制約、および妥当性検査規則のうちの少なくとも１つを表す、請求項２に記載の方法。
【請求項４】前記１つまたは複数のモーダル性特定のレンダリングが、前記アプリケーションの一部のスピーチ・ベースの表現を含む、請求項１に記載の方法。
【請求項５】前記スピーチ・ベースの表現がVoiceＸＭＬに基づく、請求項４に記載の方法。
【請求項６】前記１つまたは複数のモーダル性特定のレンダリングが、前記アプリケーションの一部の視覚ベースの表現を含む、請求項１に記載の方法。
【請求項７】前記視覚ベースの表現がＨＴＭＬ、ＣＨＴＭＬ、ＷＭＬのうちの少なくとも１つに基づく、請求項６に記載の方法。
【請求項８】前記ユーザ対話が前記対話ベースのプログラミング・コンポーネントによって宣言的に表現される、請求項１に記載の方法。
【請求項９】前記ユーザ対話が前記対話ベースのプログラミング・コンポーネントによって命令的に表現される、請求項１に記載の方法。
【請求項１０】前記ユーザ対話が前記対話ベースのプログラミング・コンポーネントによって宣言的かつ命令的に表現される、請求項１に記載の方法。
【請求項１１】前記対話ベースのプログラミング・コンポーネントが、前記ユーザと前記１つまたは複数のコンピュータ・ベースの装置との間で発生する可能性のあるダイアログに関連する基本要素を含む、請求項１に記載の方法。
【請求項１２】前記対話ベースのプログラミング・コンポーネントが複合要素を含み、前記複合要素は、前記ユーザと前記１つまたは複数のコンピュータ・ベースの装置との間で発生する可能性のあるダイアログに関連する２つ以上の前記基本要素の集合体である、請求項１１に記載の方法。
【請求項１３】前記対話ベースのプログラミング・コンポーネントの１つが会話型ジェスチャを表す、請求項１に記載の方法。
【請求項１４】前記会話型ジェスチャが、ユーザへの情報メッセージをカプセル化するジェスチャを含む、請求項１３に記載の方法。
【請求項１５】前記会話型ジェスチャが、コンテキスト・ヘルプ情報をカプセル化するジェスチャを含む、請求項１３に記載の方法。
【請求項１６】前記会話型ジェスチャが、別のジェスチャの完了が成功したときに行われるアクションをカプセル化するジェスチャを含む、請求項１３に記載の方法。
【請求項１７】前記会話型ジェスチャが、イエス／ノー・ベースの質問をカプセル化するジェスチャを含む、請求項１３に記載の方法。
【請求項１８】前記会話型ジェスチャが、ユーザが選択肢のセットから選択することを期待される場合のダイアログをカプセル化するジェスチャを含む、請求項１３に記載の方法。
【請求項１９】前記選択ジェスチャが前記選択肢のセットを表すサブ要素を含む、請求項１８に記載の方法。
【請求項２０】前記選択ジェスチャが、選択がパスすべきテストを表すサブ要素を含む、請求項１８に記載の方法。
【請求項２１】前記選択ジェスチャが、前記テストが不合格の場合に呈示すべきエラー・メッセージを表すサブ要素を含む、請求項２０に記載の方法。
【請求項２２】前記会話型ジェスチャが、所与の会話型ジェスチャの結果を妥当性検査するための規則をカプセル化するジェスチャを含む、請求項１３に記載の方法。
【請求項２３】前記会話型ジェスチャが、文法処理規則をカプセル化するジェスチャを含む、請求項１３に記載の方法。
【請求項２４】前記会話型ジェスチャが、ユーザが前記アプリケーションの各部分をナビゲートするのを助けるダイアログをカプセル化するジェスチャを含む、請求項１３に記載の方法。
【請求項２５】前記会話型ジェスチャが、少なくとも１つのユーザ・ログインおよび認証の情報を求める要求をカプセル化するジェスチャを含む、請求項１３に記載の

方法。
【請求項２６】前記会話型ジェスチャが、制約付きのユーザ入力を求める要求をカプセル化するジェスチャを含む、請求項１３に記載の方法。
【請求項２７】前記会話型ジェスチャが、制約のないユーザ入力を求める要求をカプセル化するジェスチャを含む、請求項１３に記載の方法。
【請求項２８】前記会話型ジェスチャが、情報のサブミットを制御するジェスチャを含む、請求項１３に記載の方法。
【請求項２９】論理入力イベント、ならびに、前記論理入力イベントと定義された前記論理入力イベントをトリガする物理入力イベントとの間の関連を定義する機構を提供するステップをさらに含む、請求項１に記載の方法。
【請求項３０】前記コンポーネントごとのトランスコーディングがＸＳＬ変換規則に従って行われる、請求項１に記載の方法。
【請求項３１】前記コンポーネントごとのトランスコーディングがJava Beanに従って行われる、請求項１に記載の方法。
【請求項３２】前記コンポーネントごとのトランスコーディングがJava Server Pageに従って行われる、請求項１に記載の方法。
【請求項３３】前記対話ベースのプログラミング・コンポーネントによるプレゼンテーションが、前記１つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの前記１つまたは複数のモーダル性特定のレンダリングを同期させることを可能にする、請求項１に記載の方法。
【請求項３４】前記対話ベースのプログラミング・コンポーネントによるプレゼンテーションが自然言語理解環境をサポートする、請求項１に記載の方法。
【請求項３５】前記１つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの前記１つまたは複数のモーダル性特定のレンダリングに関連するプレゼンテーション・フィーチャを表面的に変更することを可能にするコードを含めるステップをさらに含む、請求項１に記載の方法。
【請求項３６】コンポーネントごとにトランスコーディングして前記１つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの前記１つまたは複数のモーダル性特定のレンダリングを生成するための規則を変更することを可能にするコードを含めるステップをさらに含む、請求項１に記載の方法。
【請求項３７】実装されている、基礎をなすデータ・モデルの定義が、前記ユーザ対話を定義するマークアップ言語から分離される、請求項１に記載の方法。
【請求項３８】node_id属性が各コンポーネントに付加され、前記属性は様々な出力の上にマッピングされる、請求項１に記載の方法。
【請求項３９】モーダル性特定のマークアップ・コンポーネントをカプセル化する通過機構が作成者に提供される、請求項１に記載の方法。
【請求項４０】前記コンポーネントが並列に活動化させることができる、請求項１に記載の方法。
【請求項４１】前記プレゼンテーションおよびトランスコーディングが拡張可能である、請求項１に記載の方法。
【請求項４２】前記アプリケーションの状態がカプセル化される、請求項１に記載の方法。
【請求項４３】前記表現が動的に生成されるデータの参照を可能にし、前記コンテンツ／アプリケーション論理へのコールバック機構をサポートする、請求項１に記載の方法。
【請求項４４】１つまたは複数のコンピュータ・ベースの装置に関連するアプリケーションにアクセスする際に使用する装置であって、
１つまたは複数のプロセッサを備え、前記プロセッサが、（ｉ）アプリケーション・サーバから前記アプリケーションを得るように動作可能であり、前記アプリケーションが、前記１つまたは複数のコンピュータ・ベースの装置とユーザが対話ベースのプログラミング・コンポーネントによって行うことが可能な対話によってプログラム的に表現され、前記対話ベースのプログラミング・コンポーネントが、前記アプリケーションに関連するコンテンツ／アプリケーション論理およびプレゼンテーション要件に対して独立し、前記プロセッサはまた、（ｉｉ）前記対話ベースのプログラミング・コンポーネントをコンポーネントごとにトランスコーディングして、前記１つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの１つまたは複数のモーダル性特定のレンダリングを生成するように動作可能である装置。
【請求項４５】前記１つまたは複数のプロセッサが前記１つまたは複数のコンピュータ・ベースの装置にわたって分散される、請求項４４に記載の装置。
【請求項４６】前記アプリケーションの少なくとも一部がサーバから、クライアントの役割を果たす前記１つまたは複数のコンピュータ・ベースの装置のうちの少なくとも１つにダウンロードされるクライアント／サーバ構成において、前記サーバに常駐する前記コンテンツ／アプリケーション論理への接続を提供するように動作可能なコードを前記アプリケーション中に含めるステップをさらに含む、請求項４４に記載の装置。
【請求項４７】前記コンテンツ／アプリケーション論理接続コードが、前記アプリケーションに関連する１つまたは複数のデータ・モデル、属性制約、および妥当性検査規則のうちの少なくとも１つを表す、請求項４６に記載の装置。
【請求項４８】前記１つまたは複数のモーダル性特定の

レンダリングが、前記アプリケーションの一部のスピーチ・ベースの表現を含む、請求項４４に記載の装置。
【請求項４９】前記スピーチ・ベースの表現がVoiceＸＭＬに基づく、請求項４８に記載の装置。
【請求項５０】前記１つまたは複数のモーダル性特定のレンダリングが、前記アプリケーションの一部の視覚ベースの表現を含む、請求項４４に記載の装置。
【請求項５１】前記視覚ベースの表現がＨＴＭＬ、ＣＨＴＭＬ、ＷＭＬのうちの少なくとも１つに基づく、請求項５０に記載の装置。
【請求項５２】前記ユーザ対話が前記対話ベースのプログラミング・コンポーネントによって宣言的に表現される、請求項４４に記載の装置。
【請求項５３】前記ユーザ対話が前記対話ベースのプログラミング・コンポーネントによって命令的に表現される、請求項４４に記載の装置。
【請求項５４】前記ユーザ対話が前記対話ベースのプログラミング・コンポーネントによって宣言的かつ命令的に表現される、請求項４４に記載の装置。
【請求項５５】前記対話ベースのプログラミング・コンポーネントが、前記ユーザと前記１つまたは複数のコンピュータ・ベースの装置との間で発生する可能性のあるダイアログに関連する基本要素を含む、請求項４４に記載の装置。
【請求項５６】前記対話ベースのプログラミング・コンポーネントが複合要素を含み、前記複合要素は、前記ユーザと前記１つまたは複数のコンピュータ・ベースの装置との間で発生する可能性のあるダイアログに関連する２つ以上の前記基本要素の集合体である、請求項５５に記載の装置。
【請求項５７】前記対話ベースのプログラミング・コンポーネントの１つが会話型ジェスチャを表す、請求項４４に記載の装置。
【請求項５８】前記会話型ジェスチャが、ユーザへの情報メッセージをカプセル化するジェスチャを含む、請求項５７に記載の装置。
【請求項５９】前記会話型ジェスチャが、コンテキスト・ヘルプ情報をカプセル化するジェスチャを含む、請求項５７に記載の装置。
【請求項６０】前記会話型ジェスチャが、別のジェスチャの完了が成功したときに行われるアクションをカプセル化するジェスチャを含む、請求項５７に記載の装置。
【請求項６１】前記会話型ジェスチャが、イエス／ノー・ベースの質問をカプセル化するジェスチャを含む、請求項５７に記載の装置。
【請求項６２】前記会話型ジェスチャが、ユーザが選択肢のセットから選択することを期待される場合のダイアログをカプセル化するジェスチャを含む、請求項５７に記載の装置。
【請求項６３】前記選択ジェスチャが前記選択肢のセットを表すサブ要素を含む、請求項６２に記載の装置。
【請求項６４】前記選択ジェスチャが、選択がパスすべきテストを表すサブ要素を含む、請求項６２に記載の装置。
【請求項６５】前記選択ジェスチャが、前記テストが不合格の場合に呈示すべきエラー・メッセージを表すサブ要素を含む、請求項６４に記載の装置。
【請求項６６】前記会話型ジェスチャが、所与の会話型ジェスチャの結果を妥当性検査するための規則をカプセル化するジェスチャを含む、請求項５７に記載の装置。
【請求項６７】前記会話型ジェスチャが、文法処理規則をカプセル化するジェスチャを含む、請求項５７に記載の装置。
【請求項６８】前記会話型ジェスチャが、ユーザが前記アプリケーションの各部分をナビゲートするのを助けるダイアログをカプセル化するジェスチャを含む、請求項５７に記載の装置。
【請求項６９】前記会話型ジェスチャが、少なくとも１つのユーザ・ログインおよび認証の情報を求める要求をカプセル化するジェスチャを含む、請求項５７に記載の装置。
【請求項７０】前記会話型ジェスチャが、制約付きのユーザ入力を求める要求をカプセル化するジェスチャを含む、請求項５７に記載の装置。
【請求項７１】前記会話型ジェスチャが、制約のないユーザ入力を求める要求をカプセル化するジェスチャを含む、請求項５７に記載の装置。
【請求項７２】前記会話型ジェスチャが、情報のサブミットを制御するジェスチャを含む、請求項５７に記載の装置。
【請求項７３】論理入力イベント、ならびに、前記論理入力イベントと定義された前記論理入力イベントをトリガする物理入力イベントとの間の関連を定義するための機構を提供するステップをさらに含む、請求項４４に記載の装置。
【請求項７４】前記コンポーネントごとのトランスコーディングがＸＳＬ変換規則に従って行われる、請求項４４に記載の装置。
【請求項７５】前記コンポーネントごとのトランスコーディングがJava Beanに従って行われる、請求項４４に記載の装置。
【請求項７６】前記コンポーネントごとのトランスコーディングがJava Server Pageに従って行われる、請求項４４に記載の装置。
【請求項７７】前記対話ベースのプログラミング・コンポーネントによるプレゼンテーションが、前記１つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの前記１つまたは複数のモーダル性特定のレンダリングを同期させることを可能にする、請求項４４に記載の装置。

【請求項７８】前記対話ベースのプログラミング・コンポーネントによるプレゼンテーションが自然言語理解環境をサポートする、請求項４４に記載の装置。
【請求項７９】前記１つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの前記１つまたは複数のモーダル性特定のレンダリングに関連するプレゼンテーション・フィーチャを表面的に変更することを可能にするコードを含めるステップをさらに含む、請求項４４に記載の装置。
【請求項８０】コンポーネントごとにトランスコーディングして前記１つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの前記１つまたは複数のモーダル性特定のレンダリングを生成するための規則を変更することを可能にするコードを含めるステップをさらに含む、請求項４４に記載の装置。
【請求項８１】実装されている、基礎をなすデータ・モデルの定義が、前記ユーザ対話を定義するマークアップ言語から分離される、請求項４４に記載の装置。
【請求項８２】node_id属性が各コンポーネントに付加され、前記属性は様々な出力の上にマッピングされる、請求項４４に記載の装置。
【請求項８３】モーダル性特定のマークアップ・コンポーネントをカプセル化する通過機構が作成者に提供される、請求項４４に記載の装置。
【請求項８４】前記コンポーネントが並列に活動化させることができる、請求項４４に記載の装置。
【請求項８５】前記プレゼンテーションおよびトランスコーディングが拡張可能である、請求項４４に記載の装置。
【請求項８６】前記アプリケーションの状態がカプセル化される、請求項４４に記載の装置。
【請求項８７】前記表現が動的に生成されるデータの参照を可能にし、前記コンテンツ／アプリケーション論理へのコールバック機構をサポートする、請求項４４に記載の装置。
【請求項８８】前記１つまたは複数のプロセッサが前記１つまたは複数のコンピュータ・ベースの装置にわたって分散され、前記アプリケーションが前記１つまたは複数のコンピュータ・ベースの装置にまたがって同期化される、請求項４４に記載の装置。
【請求項８９】前記アプリケーションの表現がさらに、１つまたは複数のモーダル性特定のマークアップ言語を介して前記１つまたは複数のモーダル性特定のレンダリングを表面的に変更することを可能にする、請求項４４に記載の装置。
【請求項９０】ユーザからの１つまたは複数のコンピュータ・ベースの装置を介したアプリケーションへのアクセスを提供する際に使用するブラウザ装置であって、コンピュータ実行可能なコードを含む機械読取可能媒体を備え、前記コンピュータ実行可能なコードが、実行時に、
アプリケーション・サーバからアプリケーションを得るステップであって、前記アプリケーションが、前記１つまたは複数のコンピュータ・ベースの装置とユーザが対話ベースのプログラミング・コンポーネントによって行うことが可能な対話によってプログラム的に表現され、前記対話ベースのプログラミング・コンポーネントが、前記アプリケーションに関連するコンテンツ／アプリケーション論理およびプレゼンテーション要件に対して独立しているステップと、
前記対話ベースのプログラミング・コンポーネントをコンポーネントごとにトランスコーディングして、前記１つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの１つまたは複数のモーダル性特定のレンダリングを生成するステップとを実施することを可能にするブラウザ装置。
【請求項９１】ユーザから１つまたは複数のコンピュータ・ベースの装置を介してアクセス可能なアプリケーションをプログラミングする際に使用する製造品であって、コンピュータ実行可能なコードを含む機械読取可能媒体を備え、前記コンピュータ実行可能なコードが、実行時に、
前記アプリケーションにアクセスするために使用される前記１つまたは複数のコンピュータ・ベースの装置とユーザが対話ベースのプログラミング・コンポーネントによって行うことが可能な対話を表現するステップを実施することを可能にし、
前記対話ベースのプログラミング・コンポーネントが、前記アプリケーションに関連するコンテンツ／アプリケーション論理およびプレゼンテーション要件に対して独立、さらに、コンポーネントごとにトランスコーディングされて、前記１つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの１つまたは複数のモーダル性特定のレンダリングを生成する製造品。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】本発明は一般に情報アクセス・アプリケーションに関し、より詳細には、そのような情報アクセス・アプリケーションで使用するための、対話ベースのマークアップ言語および対話ベースのマークアップ言語をサポートするマルチモーダル・ブラウジング機構に関する。
【０００２】
【従来の技術】関連出願の相互参照
本発明は、１９９９年１０月１２日出願の米国仮特許出願第６０／１５８７７７号への優先権を主張し、この開示を参照により本明細書に組み込む。本明細書は、（ｉ）ＵＳ９９／２３００８（整理番号ＹＯ９９８－３９２）として識別される１９９９年１０月１日出願のＰＣＴ国際特許出願、（ii）ＵＳ９９／２２９２７（整理

番号ＹＯ９９９－１１１）として識別される１９９９年１０月１日出願のＰＣＴ国際特許出願、（iii）ＵＳ９９／２２９２５（整理番号ＹＯ９９９－１１３）として識別される１９９９年１０月１日出願のＰＣＴ国際特許出願に関係し、上記の各ＰＣＴ国際特許出願は、１９９８年１０月２日出願の米国仮特許出願第６０／１０２９５７号および１９９９年１月２７日出願の米国仮特許出願第６０／１１７５９５号への優先権を主張する。さらに本明細書は、（iv）２０００年２月１８日出願の米国特許出願第０９／５０７５２６号（整理番号ＹＯ９９９－１７８）に関係し、この出願は、１９９９年４月７日出願の米国仮特許出願第６０／１２８０８１号および１９９９年１０月１２日出願の米国仮特許出願第６０／１５８７７７号への優先権を主張する。上に参照した関連出願のすべてを参照により本明細書に組み込む。

【０００３】様々なタイプおよび量の情報の使用可能度が劇的に上昇し、そのような情報にアクセスする従来の機構の時間または使用可能度あるいはその両方が急激に減少したために、個人は現在、いつどんな装置からでも、どんな情報に対してもアクセスまたは作用または変換できるようになりたい、あるいはそのすべてを行えるようになりたいと思っている。例えばインターネットの場合、様々な大量の情報が利用可能だが、インターネットは従来、ＨＴＴＰ（HyperText Transport Protocol）ネットワークの最上部にあるＨＴＭＬ（HyperText Mark up Language）ブラウザを使用して情報にアクセスする装置しか主にサポートしていなかった。これは、ＴＣＰ／ＩＰ（Transmission Control Protocol/Internet Protocol）の最上部に提供されていた。

【０００４】この問題に対する解決法は、そのような情報にアクセスするのに使用されるアプリケーション・プログラムを書き直し、それによって他の方法でその情報にアクセスできるようにすることが中心であった。ある解決法は、ＷＡＰ（WirelessApplication Protocol）の開発へとつながった。http://www.mobilewap.comを参照されたい。ＷＡＰは、無線ネットワーク用のＨＴＴＰに相当する。無線ネットワーク用のＨＴＭＬに相当するＷＭＬ（Wireless Markup Language）が開発された。したがって、ＨＴＴＰの最上部でＨＴＭＬが使用される方式と同様に、ＷＡＰの最上部でＷＭＬが使用される。ＷＡＰおよびＷＭＬにより、ユーザは、制約された画面レンダリングおよび限られた帯域幅の接続能力を有するセルラー・ホンを介してインターネットにアクセスすることができる。ＣＨＴＭＬが、この空間を対象とするＭＬ（マークアップ言語）のもう１つの例である。

【０００５】次に、より最近になり、ウェブ・プログラミング・モデル（ファット・クライアント・プログラミング・モデルとも呼ばれる）をボイス・アクセス、特に電話アクセスおよび自動音声応答（ＩＶＲ）システムに導入するための機構が開発された。このような機構は、通常、スピーチ・ブラウザ（またはボイス・ブラウザ）と呼ばれる。このようなスピーチ・ブラウザは、上に参照した、米国出願第６０／１０２９５７号（整理番号ＹＯ９９８－３８２）として識別される米国仮特許出願に記載されている。スピーチ・ブラウザは、VoiceＸＭＬと呼ばれる、ＸＭＬ（Extensible Markup Language）のスピーチ・ベースの変形を使用することができる。http://www.voicexml.orgを参照されたい。スピーチ・ブラウザはまた、ＷＡＰプロトコルの最上部でＷＭＬデータの交換と共に動作することもできる。

【０００６】しかし、このような手法は、アプリケーション・プログラマがマルチチャネル・サポートを提供したい場合、すなわち、ウェブ・ブラウザ（ＨＴＭＬブラウザ）、電話（ボイス・ブラウザ）、および無線ブラウザ（ＷＭＬ）、または前述の開示に定義されるマルチモーダル／会話型ブラウザへのアクセスを提供したい場合に、アプリケーション・プログラマにいくつかの問題を提起する。第１に、この手法によれば、アプリケーション・プログラマは、アプリケーションを開発するときに少なくとも３つの異なる言語、例えばＨＴＭＬ、ＷＭＬ、およびVoiceＸＭＬを扱わなければならない。すなわち、ユーザが従来の電話を介したスピーチ・ブラウザを介して、あるいはＷＡＰブラウザを使用するまたは従来のウェブ・ブラウザを使用する無線接続を介してインターネット・ベースの情報にアクセスしようとするため、アプリケーションを書くときにＨＴＭＬ、ＷＡＰ、およびVoiceＸＭＬを使用しなければならないという事実を、アプリケーションは考慮しなければならない。これは、アプリケーション開発者にとって非常に厄介なものとして知られている。第２に、この手法によれば、マルチモーダル・アプリケーション、例えばアプリケーションにアクセスするために利用されるブラウザとの視覚とスピーチの両方に基づくユーザ対話を可能にするアプリケーションを同期させるのに適した方法がない。

【０００７】アプリケーションは、従来、コンテンツ（すなわち情報またはその他のデータ）とプレゼンテーション（すなわちコンテンツがユーザに呈示される方式）の両方が混合されるように開発されてきた。しかし、アプリケーション・プログラミングを簡単にする試みにおいて、コンテンツをプレゼンテーションから分離する努力がなされてきた。こうしてＸＳＬ（Extensible Stylesheet Language）が開発されたが、これは、アプリケーションに関連するコンテンツがＸＭＬで記憶され、特定の装置上でコンテンツを呈示するのに必要な変換がＸＳＬによって扱われるように、ＸＭＬと共に動作する。http://www.w3.org/Style/XSLを参照されたい。このような手法は、Ｗ３Ｃ（ワールド・ワイド・ウェブ・コンソーシアム）によって採用されている。この手法は、メイン・ブラウザ（例えば異なるバージョンのMicrosoft Internet Explorer、Netscape Communicator/Nav

igator、その他のより一般的でないブラウザなど）の特徴にプレゼンテーションを適合させるのに通常使用される。この使用を他のモーダル性／チャネル（例えば、組込み装置（無線電話またはＰＤＡ）の最上部にあるＷＭＬのようなフォーマットをサポートする無線ブラウザ）に拡張しようとする者もいた。この最後の手法はあまり成功しておらず、または便利になっておらず、いずれの場合でも、これは、ＸＳＬページを複数回オーサリングすること必要とする。しかし、この手法は、アプリケーションと装置／チャネルの両方に依存するという欠点を有する。すなわち、ＸＳＬ規則は、コンテンツを転記すべきアプリケーションおよび装置に依存する。したがって、アプリケーションが新しい装置からアクセスされる場合、その装置用に新しいＸＳＬ変換を書かなければならない。

【０００８】これらの問題のいくつかを克服するための他の試みもなされてきた。ユーザの意図（複雑かつタスク指向の意図）に基づいたＸＭＬモデルを提供する試みがあった。ユーザの意図は複雑なコンポーネントでモデリングすることができるが、これは、小さなスクリーンまたはスピーチで装置上にレンダリングすることができないかまたは非常に難しい。それより小さい原子コンポーネントに分解されないこれらの複雑なコンポーネントはまた、モーダル性にまたがって密に同期させることができない。異なるブラウザによってレンダリングされる、装置独立のタグが提供される。また、スピーチ自動音声応答（ＩＶＲ）システムへのいくつかの拡張が提案されてきた。しかし、欠陥の中でもとりわけ、これらの試みはダイアログをモデリングせず、モーダル性からモーダル性へのトランスコーディングは一般に不可能な作業である。

【０００９】これらの手法では、ユーザの意図は、複雑な対話を記述する複雑なコンポーネントでモデリングされる。しかし、これらは通常、アプリケーション特定である。すなわち、これらは、ビジネス論理の概念および要素に依存し、またはそれを特徴づけ、または必要とする。したがってその場合、ＸＳＬ規則（およびＸＳＬスタイル・シート）が今日、根本的にアプリケーションまたはアプリケーション領域（すなわち関係するＸＭＬ属性の性質）の関数であるのと同じ形で、これらの言語で書かれたページを変換するのに使用されるＸＳＬ規則もまた、根本的にアプリケーションまたはアプリケーション領域の関数である。これらは、新しいアプリケーションごとに書き直さなければならない。これは、これらの手法の限界を特徴づけている。これらの手法は、アクセスのモーダル性に対して独立したコンテンツへのアクセスを提供するのを助ける際に貢献しない。実に、これらの手法は、そのアプリケーションまたはアプリケーション領域に関係するコンテンツへのアクセスを可能にするだけである。他のどんな場合も、変換規則の書直しを必要とする。したがって、バックエンド・アプリケーションから変換規則を解放し、それをアクセス装置またはチャネルによってサポートされる特徴／モーダル性だけに依存させることが必要とされている。

【００１０】いくつかの場合に、複数のチャネルのサポートは、スタイル・シートのカスケードを使用して、得られたＸＭＬストリームをシリアル化された内部ＡＰＩ（Application Programming Interface）として扱うことによって達成されてきたことに留意されたい。この場合もやはり、これは複数のオーサリングを必要とする。

【００１１】さらに、上記の手法は、ＷＭＬのようなモーダル性で適切な対応するレンダリングを有しないコンポーネントによる、非常に複雑な意図モデルを有することになる。これらのモデルが、異なるタイプの（本質的に同じチャネルおよびモーダル性の変形内の）表示装置またはブラウザの要件にグラフィカル・ユーザ・インタフェース（ＧＵＩ）プレゼンテーションをカストマイズする能力を提供するために設計されたことは明らかである。この結果、これらの手法はどれも、スピーチまたはマルチモーダルのユーザ・インタフェースを適切にモデリングせず処理しない。

【００１２】すでに述べたように、従来のトランスコーディング（ＸＭＬコンテンツを呈示するのに使用されるＸＳＬ規則、およびあるモーダル性から別のモーダル性に移るためのＸＳＬスタイル・シートの変更）は、異なるアクセス・モーダル性をサポートすると考えられてきた。これは、所与のＸＭＬコンテンツに対し、システムが、ＸＭＬ規則を変更することによってＨＴＭＬページ、ＷＭＬ規則、またはVoiceＸＭＬページやその他さえも作成することができることを意味する。実際、これは今日、市販の異なるウェブ・ブラウザ、例えばNetscape Communicator、Microsoft Internet Explorer、Sun Microsystems Hot Java、Spyglassブラウザ、Open Source Amayaブラウザ／エディタなどをサポートするのに使用されているものである。残念ながらこれは、以下の場合のみに可能である。

（ｉ）ＸＳＬ規則がアプリケーションまたはアプリケーション領域（すなわちＸＭＬ属性の性質）に特定であり、

（ii）トランスコーディングが２つの言語間、例えばＨＴＭＬとＷＭＬとの間であり、元のコンテンツが非常に厳格なオーサリングの規則を守りながらＨＴＭＬで構築された場合。もっとも、これは、所与の会社内の所与のウェブ・サイトに対する場合にのみ施行可能である。そのような場合でも、これは、一般にはほとんど実施不可能である。というのは、他のモーダル性で対応するコンポーネントを提供するための、マークアップ言語またはモーダル性にまたがる情報の欠落（例えば、ＨＴＭＬフォームまたはメニューが音声によって自動的にそれをレンダリングするために必要な情報を提供しない）、なら

びに異なるモーダル性におけるダイアログ・ナビゲーション・フローの違いのせいである。
【００１３】したがって、既存の言語およびブラウザにあるとされるこれらおよび他の欠点を克服する、アプリケーション・プログラミング言語およびそれに関連する情報ブラウジング機構が必要とされている。
【００１４】
【発明が解決しようとする課題】本発明は、ユーザが任意のタイプの情報にアクセスするために利用している任意の装置とのユーザ対話に基づく新しいアプリケーション・プログラミング言語を提供する。この新しい言語を、本明細書では「会話型マークアップ言語（Conversational Markup Language（ＣＭＬ））」と呼ぶ。
【００１５】
【課題を解決するための手段】好ましい一実施形態では、ＣＭＬは、ユーザが所与のコンピューティング装置と行うことになる「ダイアログ」または「会話」を表すためのハイレベルＸＭＬベースの言語である。ダイアログおよび会話という用語を本明細書で使用するが、これらはより一般的に、モーダル性および装置に対して独立した、装置（ローカル装置、リモート装置（例えば電話を介した対話）、または他の方法で分散されたいずれかの装置）とのユーザ対話を指すことを理解されたい。したがって対話は、これらに限定されないが、視覚ベース（テキストまたはグラフィカル）のユーザ対話およびスピーチ・ベースのユーザ対話、ならびにこれらの組合せを含むことができる。
【００１６】このような言語により、アプリケーション作成者は、本明細書で以後「会話型ジェスチャ（conversational gesture）」と呼ぶ対話ベースの要素を使用してアプリケーションをプログラムすることができる。会話型ジェスチャは、それに従ってプログラムされたアプリケーションに関連する情報にアクセスするために利用されるモーダル性、装置、またはブラウザに対して独立してどんなダイアログも記述する、ＣＭＬの基本的なプログラミング・コンポーネントまたは要素である。
【００１７】本発明は、新しいアプリケーション・プログラミング・パラダイムを提起することにより、これらおよび他の特徴および利点を達成する。前述のように、既存のアプリケーション・オーサリング手法は、アプリケーションのコンテンツ・ベースのアスペクトをプレゼンテーション・ベースのアスペクトから分離する概念を採用してきた。本発明によれば、ＣＭＬは、アプリケーション・プログラミングをコンテンツ・アスペクト、プレゼンテーション・アスペクト、および対話アスペクトに分離することを可能にする新しいパラダイムを導入する。ユーザに関してアプリケーションの対話アスペクトに焦点を当てることにより、アプリケーションは、コンテンツ／アプリケーションの論理およびプレゼンテーションに対して独立した形で書くことができる。アプリケーションのコンテンツ論理またはビジネス論理、あるいはその両方は、アプリケーションに関連する「バックエンド論理」とも呼ぶことを理解されたい。
【００１８】クライアント／サーバ構成では、「バックエンド論理」は、論理すなわち、アプリケーションの進化を促す状態および状況のコード化されたセット、ならびに可変の妥当性検査情報を含むアプリケーションの部分である。追って説明するが、バックエンド・データから分離された論理情報を伝えるために、属性制約および妥当性検査情報をＣＭＬページに加えることができる。したがって、以下に説明し例示するが、アプリケーションがＣＭＬで作成された後、そのアプリケーションに関連するＣＭＬコードの一部がサーバからクライアント装置にダウンロードされ、次いで、ＣＭＬコードのＣＭＬジェスチャが、その装置で利用されるブラウザ特定のマークアップ言語、例えばＨＴＭＬまたはVoiceＸＭＬ、あるいはその両方にトランスコーディングされる。
【００１９】本発明によれば、ダウンロードされたＣＭＬコードで動作する装置（クライアント、さらにはＣＭＬページをＨＴＭＬ、VoiceＸＭＬ、ＷＭＬなど、おそらく他のレガシー・マークアップ言語にサービスするサーバ）は、例えばＨＴＭＬおよびVoiceＸＭＬにほぼ同時にトランスコーディングし、したがってユーザに情報へのアクセスを提供する複数のブラウザを同期させることができる。本発明によるこのような有利な同期化は、トランスコーディングがジェスチャ識別によってジェスチャごとに行われるために可能である。したがって、あるモーダル性で入力／出力イベントが発生するとき、ブラウザは、どんなイベントがどんなジェスチャに対して発生したかを知り、サポートされるすべてのモーダル性を即座に更新することができる。これにより、モーダル性にまたがる非常に密な同期化がもたらされる。このような同期化はまた、例えばグラフィカル・ユーザ・インタフェース（ＧＵＩ）ブラウザまたはスピーチ・ブラウザに関連する、モーダル性特定の様々なユーザ・インタフェース・ダイアログが、ジェスチャごとに単一のＣＭＬ表現から生成されるという事実によっても達成される。したがって、ユーザが一方または他方のモーダル性を対話的に続行するのに伴い、複数のユーザ・インタフェース、例えばＧＵＩ、スピーチなどが同期化され、連続的に更新される。本発明のＣＭＬおよびブラウジング機構は、自然言語（ＮＬ）プログラミング用のプラットフォームも提供する。ＣＭＬによってアプリケーション作成者がジェスチャごとにプログラムすることができるため、そのようなアプリケーションは、広範囲の自然会話方式で要求／応答を提供するフレキシビリティをユーザに提供する。したがってユーザは、単一のコマンドに制約されるのではなく、より制約されない形で、例えばより自然の会話に近い形で、アプリケーションと対話することができる。ＮＬおよび本発明によって、ユーザ

は、もう１人の人間と行っているかのような自然な会話を行う以外にどんな制約もなく、複数のモーダル性で自由に自身を表現することができる。ＮＬの場合にはさらに、システムは、コンテキストおよび過去の対話／ダイアログ履歴（ならびにユーザ・プリファレンス、アプリケーション設定、記憶された共通知識などの他のメタ情報）を使用して、照会を一義化することができる。

【００２０】ＮＬは、スピーチに限定されず、自然マルチモーダル会話型アプリケーションのすべてのアスペクトを含むステートメントである。これは、ＮＬ入力を自然マルチモーダル入力と結合する。上に参照した整理番号ＹＯ９９９－１１１によって識別される特許出願に記載のように、どんな入力も、モーダル性に対して独立して入力／出力イベントとしてモデリングされ、次いでそれは、ダイアログ・マネージャおよびアービトレータによって処理され、ダイアログ・マネージャおよびアービトレータは、履歴、ダイアログ・コンテキスト、および他のメタ情報（例えばユーザ・プリファレンス、装置およびアプリケーションの情報）を使用して、入力イベントのターゲットを決定する、あるいは要求されたアクションを実行する前にダイアログをユーザに関係させてユーザの意図を完結、確認、訂正、または一義化する、あるいはその両方を行うことになる。

【００２１】本発明がマルチデバイスまたは分散ブラウジング環境を可能にすることも理解されたい。複数のブラウザを効果的に同期させるＣＭＬの性質およびその能力により、アプリケーションの様々な部分を別々のコンピューティング・デバイス上に常駐させて実行することができる。次いでユーザは、あるアプリケーションにアクセスするときに複数の装置、例えばラップトップ・コンピュータおよびセルラー・ホンと同時に対話することができる。これを「マルチデバイス・ブラウジング」と呼ぶ。実際、本発明のこの態様は、「マルチモーダル性」を必要としない。すなわち、ＧＵＩ／ＨＴＭＬブラウザだけでも、ジェスチャ・ベースのＸＳＬ規則を使用して、どのブラウザに何がレンダリングされるかを定義することができる。したがって、いくつかのコンテンツを携帯情報端末（personal digitalassistant）すなわちＰＤＡ上に表示し（すなわちカラー画像、ストリーミングされたビデオ、長いリスト）、残りのコンテンツをセルラー・ホン画面上などに表示することができる。

【００２２】ＣＭＬがモーダル性独立なので、アプリケーションが書かれた後でも、どんなタイプのブラウザに関連するどんなトランスコーディング規則も実施することができる。すなわち、ＣＭＬにより、作成者は、元々実装されていた可能性のあるデフォルトのトランスコーディング以外の別のタイプのトランスコーディング（すなわちジェスチャ・ベースのトランスコーディング規則）に変更することができる。したがって、この特徴は有利にも、ジェスチャ・ベースのＸＳＬ規則の単純な更新によって、いわゆる「レガシー言語」、例えばＨＴＭＬ、ＷＭＬ、ＶoiceＸＭＬなどの新しいリリース／バージョンと、新しい言語、例えばＣＨＴＭＬ、ＨＤＭＬなどに対するサポートを保証する。さらに、この特徴は、単純なジェスチャ・ベースのＸＳＬ規則を使用して、あるバージョンのＣＭＬから新しいＣＭＬへの単純かつ容易な通路を可能にする。バージョンからバージョンへのジェスチャごとのトランスコーディングは、ＣＭＬから他のレガシー言語へのトランスコーディングと異なる問題ではないことを理解されたい。これは、ＣＭＬが定義によってこのトランスコーディングの原理を取り巻くように設計されるので、特に有利である。これが他のマークアップ言語のほとんどに当てはまらないことは確かであり、他のマークアップ言語では、後方互換性は提供するかもしれないが、仕様の更新は通常、新世代ブラウザに対して、ならびにより古いバージョンで書かれたより古いすべてのコンテンツに関して問題を含む。

【００２３】ＣＭＬはまた、ＣＭＬページが書かれた後でも、プレゼンテーションを表面的に変更することを可能にする。例えば、所望のモーダル性およびターゲット・マークアップ言語に応じてＣＭＬコマンドを発行して、いくつかのモーダル性におけるコンテンツのプレゼンテーションの、いくつかのフィーチャを表面的に変更することができる。これにより、ＣＭＬ開発者は、ＨＴＭＬレンダリングに対してするのと同じ量の表面的努力で済む。しかし、利点は当然、（アクセス装置またはチャネルに対して独立）ユニバーサル・アクセスを提供するのに使用できる、またはマルチモーダルおよび会話型ユーザ・インタフェースを密に同期させることができる、あるいはその両方ができる、対話のマルチチャネル（すなわち複数のタイプのターゲットＭＬまたはデバイス・モーダル性または特定のユーザ・インタフェース特徴で表現することができる）記述を、同じ代償で得たことである。

【００２４】本発明はまた、様々なモーダル性特定の表現、例えばＨＴＭＬベースのグラフィカル・ユーザ・インタフェース（ＧＵＩ）ブラウザ、ＶoiceＸＭＬベースのスピーチ・ブラウザなどに従ってＣＭＬの機能をサポートすることのできるマルチモーダル・ブラウザの様々な実施形態を可能にする。

【００２５】用語「ＣＭＬ」は、上で参照した整理番号ＹＯ９９８－３９２およびＹＯ９９９－１７８によって識別される特許出願で使用されていることに留意されたい。これらの出願では、この用語は、会話型インタフェースを記述するための宣言的な方式を指すものとされる。本発明によれば、用語ＣＭＬは、以下に詳細に述べるように、対話によるプログラミングの概念を組み入れたジェスチャ・ベースの言語について言う。

【００２６】本発明のこのような態様ならびに以下に述べる他の態様があった場合に、ここで、このような創意

に富んだ特徴と既存の手法との重要な違いをいくつか考察する。この５年間のワールド・ワイド・ウェブ（ＷＷＷ）の急激な成長は、まずコンテンツからユーザ対話を分離し、続いてプラットフォームに依存するＷＷＷブラウザによってレンダリングされるＨＴＭＬのようなマークアップ言語を介してアプリケーション・フロント・エンドを送達することによって軽量ユーザ・インタフェース・アプリケーションを構築することにおける潜在的な強さを指摘した。この体系は、基礎をなすハードウェアおよびオペレーティング・システムの細部からエンドユーザ・アプリケーションを解放することによって、新しい可能性の世界を開く。現在のＷＷＷ体系は、基礎をなすハードウェアおよびオペレーティング・システムの細部から、ｅコマース・アプリケーションへの視覚インタフェースを開放した。この進化における次のステップは、インタフェース・モーダル性と、電子情報と対話するのに使用される装置とに対して独立したエンドユーザ・アプリケーションを作ることである。この進化は、新世代のｅコマース・アプリケーションとのスピーチ・ベースの対話を可能にすることにおいて自然な次のステップである。

【００２７】装置およびモーダル性から独立したエンドユーザＷＷＷサービスを達成するために、様々な装置への送達を可能にするような、モーダル性独立の技術を使用するアプリケーションおよびサービスをオーサリングすることが強く必要とされている。ＸＭＬが急速にＷＷＷの次世代共通語となっていることから、そのような言語をＸＭＬアプリケーションとして設計することが自然である。

【００２８】したがって、モーダル性独立の情報コンテンツおよび対話論理をオーサリングするためのＸＭＬベースの言語を設計し、次いで、得られたアプリケーションをターゲット装置に最適な仕方で送達することにより、モーダル性独立のＷＷＷサービスを達成することができる。これは必然的に、情報コンテンツ、情報プレゼンテーション、および対話論理を別個のコンポーネントに分離する言語を設計することになる。ＷＷＷはすでに、スタイル・シートを利用することによってコンテンツをプレゼンテーションから分離することに向けて進化している。次の進化ステップは、対話論理を情報コンテンツから抽出することである。現在、この領域での外部規格活動が、Ｗ３Ｃなどの産業コンソーシアムのＸＦＯＲＭＳおよびボイス・ブラウザの委員会内から現れることが予想されている。

【００２９】上に概説した分離は、会話型コンピューティングと我々が呼ぶ手法となる。エンドユーザ・アプリケーションおよびサービスは、モーダル性独立の会話型ジェスチャの集合として表され、各会話型ジェスチャは、ユーザ対話を構成するマン・マシン・ダイアログの原子的な一片をコード化する。

【００３０】上に概説した洞察は、意図ベースのマークアップ言語を設計する試みが近い過去にわずかにあったという事実によって妥当性検査される。これらは最初に、異なる装置間、例えば小さい画面のハンドヘルド対デスクトップＰＣの視覚プレゼンテーションにおける違いを抽出するように設計された。スピーチ・インタフェースが関係してくるとき、これらの言語は両方とも、これらの元のターゲットであった異なる視覚表示に加えてスピーチ装置に送達するためのエンドユーザ・アプリケーションをオーサリングするための可能な手段として呈示される。

【００３１】本発明によれば、ＣＭＬは、スピーチおよび自然言語技術を含む会話型インタフェースに対処する必要性によって導入される新たな要件に特に焦点を当てられて、モーダル性独立のユーザ対話に向けたＸＭＬベースの言語として最初から設計される。ユーザ・インタフェースにおける一流市民としてのスピーチへのこの焦点は、ＣＭＬを従来の試みとは異なる形で進化させた。これらの、鍵となる違いのいくつかを対比することにする。

【００３２】（ｉ）データ・モデル上への対話のオーバーレー

従来技術の言語はすべて、ユーザの意図と、ユーザ対話によって同じ一片のマークアップ内に配置される、基礎をなすデータ・モデルとを定義する。ある仕様からこれを例示する短い例を挙げる。以下に示すマークアップの断片は、人の称号（Ｍｒ．、Ｍｒｓ．、Ｍｓ）を得るのに使用されることになる。プロンプト指示されるデータ・モデルの定義が、ユーザ対話を生成するマークアップと混ざっていることに留意されたい。

```
<prior art ML>
        <CHOICE          NAME="Person Titles"
SELECTION-POLICY="SINGLE">
            <CAPTION>Title</CAPTION>
            <HINT>This is a set of valid titles for a person.</
HINT>
            <STRING NAME="Mr">
                <VALUE>Mr.<VALUE>
            </STRING>
            <STRING NAME="MRS">
```

```
        <VALUE>Mrs.<VALUE>
    </STRING>
    <STRING NAME="MISS">
        <VALUE>Miss</VALUE>
    </STRING>
    <STRING NAME="MS">
        <VALUE>Ms<VALUE>
    </STRING>
</CHOICE>
</prior art ML>
```

【００３３】上記を、以下に示す、人の称号を得るためのＣＭＬ表現と比較されたい。データ・モデルの定義、すなわち有効な人の称号をリストする列挙型を、ユーザ対話コンポーネントすなわち選択ジェスチャから分離していることに留意されたい。

【００３４】まず、列挙型PersonTitleを定義する。

```
<enum name="PersonTitle"type="string"> <value>MR</value> <value>MRS</value> <value>MISS</value> </enum>
```

【００３５】フィールドPersonTitleが定義されると、それは、ユーザ対話中の複数のポイントで適切なＣＭＬジェスチャを介してインスタンス化することができる。これを、ジェスチャselectによって以下に示す。

```
<select name="PersonTitle"selection-policy="single"> <message>PersonTitle</mesage> <choices> <choicevalue="MR">Mr.</choice>  <choicevalue="MRS>MRS.</choice> <choicevalue="MISS">Miss.</choice>  </choices> </select>
```

【００３６】基礎をなすデータ・モデルの定義（上の列挙PersonTitle）から会話型ジェスチャ（上の例ではジェスチャselect）を分離することは、次のようないくつかの利点をもたらす。

【００３７】（１）データ定義から会話型ジェスチャを分離することにより、例えば上のダイアログを国際化するときに、人の称号をプロンプト指示するための複数のユーザ・インタフェースをオーサリングすることができる。したがって、ＣＭＬで構成されたこのダイアログのドイツ語バージョンは、会話型ジェスチャを修正されることだけしか必要としないことになる。上記の表現が国際化されるとき、すなわち変更が必要なのは要素caption、hint、およびcode valueのコンテンツであるとき、基礎をなす列挙型の定義は同じままであることに留意されたい。しかし、ユーザ・インタフェース・マークアップをデータ定義上にオーバーレーすることによれば、この設計は、ダイアログを国際化するのに必要な変更を切り離すことができない。従来の言語のいくつかは、上記のダイアログの異なる言語バージョンを作成するために作成者によって次いで再利用されるテンプレートの概念を導入することによって、国際化に関するこの明示的な問題を回避していることに留意されたい。しかしこれは、根源にある基本的な問題を除去していない。すなわち、データ定義およびユーザ・インタフェースは、依然としてテンプレート定義中でリンクされたままである。

【００３８】（２）フィールドPersonTitleが定義されると、ＣＭＬジェスチャは、ユーザ対話中の複数のポイントでこのフィールドを参照することができる。したがって、ユーザがフィールドPersonTitleに対する値を指定すると、ダイアログの後続の部分は、プロンプト、例えばWelcome to the electronic store Mr. Smithを作成するときに、供給された値を参照することができる。

【００３９】（３）ＣＭＬでオーサリングされたアプリケーションはまた、ユーザ対話の異なるポイントでPersonTitleなどの特定のフィールドに対してユーザに自由にプロンプト指示することができ、ユーザは、どのポイントで自分がそのフィールドに値を供給するかを自由に決定することができる。このような形のフレキシビリティは、自然言語インタフェースの設計では特に不可欠であり、この場合もまた、ユーザ対話を宣言するマークアップからモデルを定義するマークアップを分離する結果となる。この分離がなければ（現時点の従来技術のように）、上記のものは、作成者にフィールドPersonTitleを複数回定義させることになる。

【００４０】上記のことを理解するために、ユーザがミューチュアル・ファンドを売買することができると同様に特定の資産の正味価値を知ることもできるミューチュアル・ファンド・アプリケーションを考察されたい。この対話を単純化したバージョンで、システムは、ユーザから２つの情報アイテムを得る必要がある。

（ａ）ユーザ・アクション、すなわち売買または正味資産評価

（ｂ）作用する資産、例えば購入資金

【００４１】上の例に自然言語インタフェースを使用するとき、ユーザは、システムから最初にプロンプト指示されるときに、実行するアクションと作用する資産のいずれか、またはおそらくその両方を指定することが等しく見込まれる。何が指定されたかに応じて、次にダイアログは、欠落した部分の情報に対してシステムがプロンプト指示する状態に移行する必要がある。あるいは、アクションと資産が両方とも指定された場合、システムは、フォーム「Would you like to action specified f

und?」の確認プロンプトを作成する必要がある。従来技術は現在、対話マークアップすなわちこの場合に要素CHOICEをデータ定義上にオーバーレーするので、アプリケーション作成者が同じフィールドの値、例えばユーザ対話中の異なるポイントにおける資産を得るためにユーザ対話を指定することは不可能となる。

【００４２】データ・モデルの上への対話のオーバーレーは、本明細書に開示する、我々の手法の新規性とプログラミング・モデルの新しいパラダイムとを特に強調する。

【００４３】（ii）アプリケーション状態をカプセル化するための明示的環境の欠如

ＣＭＬでユーザ対話からデータ・モデルを分離することの別の結果は、ＣＭＬ文書としてオーサリングされたアプリケーションが、アプリケーション状態、例えば上に挙げた例の中のPersonTitleまたはアクションを結合する環境を明瞭に呈示することである。従来技術の場合では、このアプリケーション状態は暗黙的であり、その言語でコード化されたユーザ・インタフェースの他の部分にとって容易に利用可能ではない。

【００４４】データ・モデルを、したがってアプリケーション状態を明示的に定義することにより、ＣＭＬは、ユーザ対話が完了するとサーバに送り返されることになるＸＭＬコード化を明瞭に定義する。したがって、フィールドPersonTitleの場合、サーバは、サブミット時に以下のものを受信することになる。

<PersonTitle>MR</PersonTitle>

【００４５】データ・モデルの定義にアクセスできるサーバは、サブミットされた値を妥当性検査することができる。より複雑な例では、データ・モデル定義は、アプリケーション特定の妥当性検査制約をカプセル化することができ、これらの制約は両方で、クライアント側でチェックして、後でサブミット時にサーバ・エンド上で妥当性検査することができる。このようにデータ・モデルおよび制約をユーザ・インタフェースから分離することにより、ユーザが特定の対話装置、例えばデスクトップＰＣを使用して対話を開始することを可能にするＣＭＬアプリケーションが、部分的に完了したトランザクションをサブミットし、後で異なる装置、例えばセルラー・ホンを使用してそのトランザクションを完了させることができる。

【００４６】（iii）従来技術はＧＵＩレガシーを反映する

従来技術の仕様に定義されているコア属性の多くは、ＧＵＩ特定のレガシーを反映している。例えば、すべてのデータ・モデルは表示されるコア属性によって制限され、これは、表示ベースのインタフェースにしか意味をなさない。スピーチ・ベースのハンドヘルドやセルラー電話などの視覚的でない装置に対し、enable=false、shown=trueなどの一義的な設定変換はないように思われる。

【００４７】さらに、これらの属性は、ユーザ対話の表現を小さいサイズの表示装置にマッピングすることを難しくする。これは、デスクトップＧＵＩ用のこれらのＭＬでオーサリングされるアプリケーションが、対話要素の多くを表示されるように宣言する見込みが高いからであり、これは表示装置のスペースが乏しい環境では難しくなる。

【００４８】従来技術は通常、大きな画面の外では意味を持たない他のＧＵＩコンポーネントを有する。残念ながら、言語内に行き渡り、モーダル性／チャネルにまたがって容易に使用可能ではない機能には問題があり、トランスコーディング／レンダリングが任意のターゲットに対して可能となることを保証することができない。

【００４９】さらに、スピーチのようなモーダル性は、ダイアログ・コンポーネントをレンダリングするために追加の情報（例えば文法、語彙、言語モデル、音響モデル、ＮＬ解析およびタグ付けデータ・ファイルなど）を必要とする可能性がある。この情報は、従来の仕掛けでは利用不可能である。この場合もまた、データ・モデルと対話の間のオーバーレーは、同じダイアログ・コンポーネントが、異なるデータ・ファイルを有するページで複数回使用されるときに問題となる。

【００５０】（iv）原子的会話型ジェスチャの欠如

従来技術によるユーザ対話の表現は、実装されている、基礎をなすデータ・モデル上に直接オーバーレーされるため、これらのＭＬには、ＣＭＬにあるような原子的会話型ジェスチャのセットの概念はない。むしろ、selectなどの明示的なＣＭＬジェスチャは、従来技術では暗黙的である。例えば、ＣＭＬジェスチャselectは、従来技術では、リスト構造に対するマークアップ上に選択要素に対するマークアップをオーバーレーした結果として表れることになる。上に挙げたフィールドPersonTitleの例を参照されたい。

【００５１】より複雑なダイアログを構成するとき、原子的会話型ジェスチャの欠如がまず問題になる。例えば、従来技術は、明示的なテーブルおよびツリー構成を導入して、２次元表レイアウトおよびツリーの仕組みのＧＵＩ概念に対応する。しかし、これらのよりハイレベルな構造はＣＭＬにおけるように原子構成単位で構築されていないため、テーブルやツリー（ここでツリーはopenまたはclosedと宣言される）のようなコンポーネント構成を、静的２次元表示が欠如したスピーチのようなモーダル性にマッピングすることは不可能である。また、ツリーやテーブルのようなジェスチャには、画面の小さい装置上で直接に相当するものがない。

【００５２】（ｖ）同期化

複数の対話モーダル性にまたがる密な同期化が、高品質なマルチモーダル・インタフェースの鍵となる要件である。さらに言えば、このようなマルチモーダル・クライ

アントは、基礎をなすプラットフォームとして従来のブラウザから提供されるＤＯＭ（Document Object Model、http://www.w3c.orgに記載）を使用して構成される見込みがより高い。この後者の実施シナリオでは、上で詳述した、ユーザ・インタフェース構成をデータ定義上にオーバーレーすることが、やはりネックになりやすい（例えば、ダイアログ／アプリケーション状態をカプセル化するための明示的な環境の欠如に対して上に述べた、この場合にビューごとの同じ問題）

【００５３】モーダル性にまたがる密な同期化は、ＣＭＬにおける基本的な目標である。これは、ＣＭＬ設計全体に反映され、会話型ジェスチャとデータ・モデルの定義との間に生じる分離は、旧来のモデル・ビュー・コントローラ（ＭＶＣ）設計を使用してＤＯＭの最上部に構築されるマルチモーダル・ブラウザの実装をより容易にする。

【００５４】（ｖｉ）会話型アプリケーション

会話型アプリケーションは、複数のフォーム（それぞれがトランザクションまたはトランザクションの一部を記述する）を同時に活動化させることによって宣言的に開発することができる。これは、ファイル中の異なる場所で同じダイアログ・コンポーネントを再利用する機能を必要とする。上に説明したように、前述のオーバーレーはこの要件をサポートしない。

【００５５】（ｖｉｉ）イベント結合の欠如

イベント結合能力の欠如は、アプリケーションのマルチチャネル／マルチモーダル機能を制限する。すなわち、何らかの具体的な論理動作を何らかの具体的な物理動作に関連付ける方式がない。これは、異なる結合が望ましいマルチモーダル／マルチチャネル・アクセス（例えば電話ヘルプ用キー・ショート・カット、ヘルプ用音声コマンド、キーボード上におけるヘルプ用のキーの組合せ）を提供したい場合にクリティカルである。

【００５６】（ｖｉｉｉ）ピア

さらに、従来技術の試みはまた、基礎をなす同じ表現から異なるユーザ対話を生成するためにピアの技術に依拠している。そうすることにより、これは、同期化されるマルチモーダル対話の問題に対処しない。

【００５７】

【発明の実施の形態】後続の説明で、好ましい仕様のＣＭＬ、好ましいマルチモーダル・ブラウジング環境、および本発明をよりよく理解するためのいくつかの例示的な適用例を使用して本発明を示す。しかし、本発明がこれらの特定の好ましい実施態様および例示的な適用例に限定されないことを理解されたい。本発明はむしろ、より一般に、アクセス・プロトコル、モーダル性、ブラウザ、または装置に関係なく、どんな情報アクセス・アプリケーションにも適用可能である。したがって、本発明はより一般に、同期化されマルチモーダルの、簡単かつ便利な情報のアクセスをユーザに提供することが望ましいどんな情報アクセス状況にも適用可能である。

【００５８】詳細な説明は、参照しやすいように次のセクションに分かれている。すなわち、（Ｉ）ＣＭＬ仕様、および（ＩＩ）ＣＭＬをサポートし、解析し、レンダリングするためのマルチモーダル・ブラウザ体系である。セクションＩは、本発明によるＣＭＬの好ましい仕様の詳細な説明を提供する。セクションＩＩは、本発明によるＣＭＬを実施するための好ましいマルチモーダル・ブラウジング環境を提供する。

【００５９】Ｉ．ＣＭＬ仕様

以下の説明は、ＣＭＬの好ましい実施形態の仕様である。このセクションは、参照しやすいように次のサブセクションに分かれている。すなわち、（Ａ）序説、（Ｂ）比較例、（Ｃ）ＣＭＬ構文、（Ｄ）名前空間、（Ｅ）ＣＭＬ属性、（Ｆ）ＣＭＬコンポーネント、（Ｇ）結合イベント、（Ｈ）ジェスチャのグループ化および定義フォーカス、（Ｉ）データ・モデルおよびデータ・モデル、（Ｊ）アクセス環境、（Ｋ）ＣＭＬ横断モデル、（Ｌ）特定ユーザ・インタフェース言語へのＣＭＬの変換、（Ｍ）表面変更、および（Ｎ）ＣＭＬ文書型定義である。

【００６０】Ａ．序説

前述のように、コンテンツ再利用を達成するためにコンテンツをプレゼンテーションから分離することは、ワールド・ワイド・ウェブ（ＷＷＷ）上で情報を展開するために従来受け入れられてきた方式である。これを図１に示す。図示のように、アプリケーション・オーサリングに関する既存の手法は、２つのコンポーネントしか考慮しない。すなわち、コンテンツ・コンポーネント（Ａ）、およびプレゼンテーションコンポーネント（Ｂ）である。現在のＷ３Ｃ体系では、このような分離は、コンテンツをＸＭＬで表現し、次いでそれをアプリケーションおよび装置に依存するＸＳＬ変換を介して適切な最終形式のプレゼンテーション（例えばＨＴＭＬ、VoiceＸＭＬ、ＷＭＬ）に変換することによって達成される。しかし、この手法にはクリティカルな欠点がある。実際、ＸＳＬ規則は通常、バックエンド・アプリケーションまたは領域に依存する。結果として、アプリケーションのオーサリングは、ＸＭＬコンテンツを設計し、次いでＸＳＬスタイル・シートをアプリケーション／ページごとに、かつターゲット装置／チャネルごとに設計することを伴う複数のオーサリング実行である。さらに、スタイル・シートを使用してあるＭＬから別のＭＬにトランスコーディングされると予想されるとき、前述のように、トランスコーディングは通常、２つのレガシー言語間（例えばＨＴＭＬからＷＭＬへ）であることが多く、次いで、非常に厳密なオーサリングの規則に従って元のコンテンツがＨＴＭＬで構築される。もっとも、これは、所与の会社内の所与のウェブ・サイトに対する場合にのみ施行可能である。そのような場合でも、

これは、一般にはほとんど実施不可能である。というのは、他のモーダル性で対応するコンポーネントを提供するための、マークアップ言語またはモーダル性の間の情報の欠落（例えば、ＨＴＭＬフォームまたはメニューが音声によって自動的にそれをレンダリングするために必要な情報を提供しない）のせいである。

【００６１】ＣＭＬは、フォーム（プレゼンテーション）およびコンテンツに加えて第３のコンポーネント、すなわち対話を実現することによって動かされ、この対話は、静的情報表現を対話型情報に変えることの中心にある。静的情報は、ユーザが受動的であり、かつ、すべての情報を呈示される、非常に特殊な場合であることを理解されたい。この新しいパラダイムを図２に示す。図示のように、本発明は、対話によるプログラミングの概念を導入し、ここで、アプリケーション・オーサリングは３つのコンポーネント、すなわちコンテンツ（Ａ）、プレゼンテーション（Ｂ）、および対話（Ｃ）に分割される。この新しいプログラミング・パラダイムは、新しいプログラミング環境、例えば開発ツールなどの開発と組になって行われる。

【００６２】この明細書を通して、少量の埋め込み型アプリケーション・インテリジェンスを有する「軽量」情報アプリケーションまたは電子情報を「インフォウェア（infoware）」と呼ぶ。現在まで、このような対話は、一部は表示的なＨＴＭＬすなわちフォーム要素内で、一部はサーブレット（servlets）およびＣＧＩ（CommonGate Interface）スクリプトにカプセル化されたサーバ側の論理内で表示されてきた。この組合せが、情報コンテンツが支配するインフォウェアすなわち軽量アプリケーションを生み出すことになった。今日のWWWにおけるインフォウェアのよい例は、Amazon.comのようなｅビジネスである。

【００６３】複数のモーダル性を介してこのようなインフォウェアと対話する世界に我々が移動するのに伴い、この３つのアスペクト、電子コンテンツすなわちコンテンツとプレゼンテーションと対話との間を明瞭に分離する時が来た。

【００６４】ＣＭＬは、すべてのマン・マシン・ダイアログが、適切なシーケンスの「会話型ジェスチャ」、または適切に結合させて任意の対話に置き換えることのできるモーダル性独立の構成単位（コンポーネントまたは要素）に分割できるという洞察に基づく。ＣＭＬは、これらの基本的な構成単位をＸＭＬでコード化することにより、モーダル性独立方式でマン・マシン対話をカプセル化する。このようなＣＭＬカプセル化は、後に適切なモーダル性依存のユーザ・インタフェースに変換される。この変換は、複数の「コントローラ」、すなわち今日のWWW中心の世界におけるブラウザの間で同期化を達成する形で行われ、これらが、単一のモーダル性独立「モデル」に対するモーダル性特定の「ビュー」を操作する。「モデル」、「ビュー」、「コントローラ」という用語は、コンピューティングの従来のＭＶＣ（モデル・ビュー・コントローラ）分解に従って使用される周知の用語であり、例えば、その開示を参照により本明細書に組み込むG.E.KrasnerおよびS.T.Popeの「A Cookbook for Using the Model-View-Controller User Interface Paradigm in SmallTalk-80」,Journal of Object-Oriented Programming,1(3):26-49, August/September 1988を参照されたい。結果として、複数の情報機器にわたって対話挙動が一貫し、複数のインタフェース・モーダル性にわたってユーザ対話が整合され正しく同期化される。

【００６５】Ｂ．比較例

ＣＭＬの好ましい実施形態の仕様を説明する前に、ＣＭＬおよび対話によるプログラミングの基本原理を示すいくつかの例を呈示する。これらの例は、「グローバル・カフェ」サイトに言及する。カフェに着く前またはカフェにいるときに顧客が自分の飲物を前もって注文することができるようにすると決定したカフェを想像されたい。したがって、彼らは基本的に、アクセス・チャネルに対して独立して自分の情報へのアクセスを提供したいと思う。

【００６６】したがって、ページがＣＭＬでオーサリングされる。このページを生成するためのＣＭＬコードをＣＭＬコード１０として図３に示す。このページは基本的に、一連の会話型ジェスチャを含む（ここでジェスチャは、ＣＭＬおよび対話によるプログラミングの基本原理がよりよく理解されるように、後で提供される実際のＣＭＬ仕様の細部からはいくぶん自由であることに留意されたい。

【００６７】このページは次のものを含むことができる。

（１）タイトル（図３で「ジェスチャ」２０として示す）：「Global Cafe」（すなわちタイトルとしてレンダリングされる特定のメッセージ）

（２）ジェスチャ・メッセージ（図３で「ジェスチャ」２２として示す）：Would you like coffee, tea, milk or nothing?

（３）ジェスチャ・リストからの排他的選択（図３で「ジェスチャ」２４として示す：リストは次のアイテムで構成される。coffee, tea, milk, nothing。

（４）サブミット・ジェスチャ（図３には明示的に示さず）

【００６８】明らかに、このページは、ターゲット・モーダル性（すなわちアクセス・チャネルまたはアクセス装置のタイプ）へのどんな依存も導入せずに、ユーザとの完全な対話を十分に定義している。このページはまた、次のような、対話によるプログラミングのプログラミング・モデルもはっきりと示している。

【００６９】（ｉ）ターゲット・モーダル性に対して独

立して対話の基本的なコンポーネントを使用して、アプリケーションが対話によってプログラムされる。
（ａ）ジェスチャ・メッセージ：「Global Cafe」
（ｂ）ジェスチャ・メッセージ：Would you like coffee, tea, milk or nothing?
（ｃ）ジェスチャ・リストからの排他的選択
（ｄ）サブミット・ジェスチャ
【００７０】（ii）これは、従来通りにプログラム／開発されるバックエンドに接続される。この例では、バックエンドへの接続はリスト（coffee, tea, milk, nothing）によって例示され、これは、ページが作成されたときには静的に、また、ページがバックエンド論理を使用してサーバ上で動的に生成されたときは動的に、バックエンド・データベース中に読み込まれ、かつリストへの引数として追加されたものである。
【００７１】（iii）この段階で必要なら、例えばＸＦＯＲＭ構文を使用して、属性／変数の制約、妥当性検査を追加することができる。例えば、ページがアルコール飲料を提供するためにユーザの年齢を要求する場合には、ユーザが自分は未成年であることを示すとダイアログを制約または修正するといった制約を容易に表すことができる。これは、このページには明示的に示していない。
【００７２】（iv）プレゼンテーションは以後、表面変更することができる。この例では、これは、ジェスチャ・メッセージの代わりにジェスチャ・タイトルを使用することによって行われる。すなわちモーダル性独立の表面変更である。モーダル性特定の表面変更もまた、例えば、得られたＨＴＭＬページに使用するための背景（色または画像）を指定するＨＴＭＬタグを追加することによって追加することができる。これは、他のターゲット・モーダル性からは無視される、あるいは他のモーダル性用に提供された「挙動」で置換されることになる。例えば、ＨＴＭＬモーダル性で画像が表示されるとき、代わりのレンダリングとなるキャプションをＷＭＬ、VoiceＸＭＬ、または他のモーダル性に提供することができる。
【００７３】（ｖ）得られたページは、次に、適切なブラウザによってレンダリングすることができる。２つのモデルが存在する。ＣＭＬページは、ＣＭＬコンテンツを解析およびレンダリングすることのできるブラウザに提供される（以下のケースＢ参照）か、またはレガシー言語、例えばＨＴＭＬ、ＷＭＬ、VoiceＸＭＬなどしか扱うことのできないレガシー・ブラウザに提供される（以下のケースＡ参照）。
【００７４】（ａ）ケースＡ：このケースは、「マルチチャネル」ケースとも呼ばれる。ターゲット・ブラウザは、要求者のアドレス（無線ゲートウェイまたはスピーチ・ブラウザ）から、または要求（すなわちＨＴＭＬファイル要求対ＷＭＬページ要求）から明確である（ＨＴＭＬブラウザ用のＨＴＴＰ接続で識別される）。ページが要求されるとき、これは、ＣＭＬで取り出され、ジェスチャ・ベースのＸＳＬ変換規則を使用して実行中にターゲットＭＬにトランスコーディングされる。
【００７５】（ｂ）ケースＢ：ターゲット・ブラウザはＣＭＬを扱う。したがって、ターゲット・ブラウザは、それがサポートするモーダル性が何であるか（単一か複数か）、ならびにサポートされるモーダル性で所与のジェスチャを最適にレンダリングするのに必要な規則を正確に知っている。これらのジェスチャＸＳＬ変換規則は、装置が構築されたときまたはブラウザがそれに移植されたときにブラウザ中にプログラムされたものであることが有利である。明らかに、これは、適切な情報を有する（すなわちその装置を十分によく理解している）最も適切なプログラマがその責務を負うことを意味する。
【００７６】（vi）いずれの場合にも、ＣＭＬアプリケーション開発者は、何もする必要がない。開発者は常に、プラットフォーム／ブラウザがレンダリングを適切に扱うと仮定することができる。
【００７７】（vii）ジェスチャは、ターゲット・モーダル性から完全に独立している。ジェスチャはまた、バックエンド・ビジネス論理／ドメインまたは他の何にも独立、ジェスチャだけに依存する。これが、ＸＳＬ規則をブラウザに記憶できる理由である。
【００７８】（viii）ＸＳＬ規則は、ターゲット・モーダル性に基づいてジェスチャをレンダリングする。この場合では、これは次のことを意味する。
（ａ）タイトル：
ＨＴＭＬ：太字の見出し文字が表示される
ＷＭＬ：単一カード表示
VoiceＸＭＬ：歓迎プロンプト
（ｂ）メッセージ：
ＨＴＭＬ：通常の文字で表示
ＷＭＬ：通常の文字で表示（おそらく複数のカード上で）
VoiceＸＭＬ：プロンプト・メッセージを生成（テキストからメッセージへ、または再生）
（ｃ）リストからの排他的選択：
ＨＴＭＬ：プルダウン・メニュー
ＷＭＬ：ラジオ・ボタン
VoiceＸＭＬ：メニュー中で選択するためのダイアログ（おそらく自然言語）（例えば、「あなたはこのように多くのアイテムから選べます。最初の３つを読みます。アイテムを選ぶか、次の３つが聞きたければもっと、と言って下さい・・・」）
【００７９】図３に戻ると、ＣＭＬで書かれたグローバル・カフェ・アプリケーションで得ることのできる３つの例示的なレンダリングを視覚化したものがある。したがって、ジェスチャ・ベースのＸＳＬ変換を含むＣＭＬコード１０から、グローバル・カフェ・アプリケーショ

ンのＨＴＭＬレンダリング１２、ＷＭＬレンダリング１４、およびVoiceＸＭＬレンダリング１６が得られる。

【００８０】（ix）マルチモーダル／会話型ブラウザによってトランスコーディングが行われるとき（以下に述べる）、ジェスチャはnode_idタグを使用して一意に識別される。これにより、登録された各モーダル性（ローカルまたは分散）でレンダリングが生成されるだけでなく、非常に密な同期化が実現する（すなわち、これが意味をなすジェスチャであるときに、ジェスチャ・レベルで、またはサブ・ジェスチャ・レベルでも）。例えば、イベント（Ｉ／Ｏイベント）は即座にダイアログの状態（すなわち、上に参照した整理番号ＹＯ９９９１７８によって識別される特許出願にあるように、例えばマルチモーダル・シェル中に維持される状態）および他のモーダル性に影響する。したがって、このような密な同期化が、携帯情報端末がサポートできるＨＴＭＬレンダリング１２と、従来の電話機がサポートできるVoiceＸＭＬレンダリング１６との間に存在することができる。

【００８１】ジェスチャＸＳＬ変換規則は、アプリケーション開発者が上書きし、それらをどこでダウンロードすべきかを示すことができることに留意されたい。それらはまた、普通ならデフォルトの挙動となるものから、ユーザ、アプリケーション、または装置のプリファレンスによって上書きすることもできる。

【００８２】新しいジェスチャを追加することもでき、その場合、関連するＸＳＬ規則を提供しなければならない（例えばそれを得る場所のＵＲＬ）。

【００８３】Ｃ．ＣＭＬ構文

ＣＭＬの好ましい実施形態では、ＣＭＬ構文はＸＭＬに準拠する。ＣＭＬインスタンスは、うまく形成されたＸＭＬである。装置の制約に基づいてＸＭＬプロセッサを妥当性検査するＣＭＬプロセッサを実施することができる。

【００８４】（ｉ）特別なＣＭＬ注記

（１）ケース依存性

ＣＭＬクライアントおよびサーバは、ＣＭＬ要素名および属性名をケース依存として扱う。慣習として、この仕様中に定義されたすべての要素名および属性名は、より低いケースを使用する。この慣習は、事前定義されたすべての要素名および属性名に厳密に課される。

【００８５】（２）コンテンツ・モデル

ＣＭＬインスタンスは、一連のＸＭＬ要素からなる。ＣＭＬは、最上レベルにあるどんなｐｃデータも許容しない。すなわち、ＣＭＬのすべての最上レベルの子は、必ず要素である。

【００８６】（３）乏しいＣＭＬ

ＣＭＬインスタンスは乏しい場合がある。属性node_idを除き、この仕様中に述べる最上レベルのＣＭＬ属性および要素が必要とされる。

【００８７】（４）エンティティ参照

ＣＭＬ中のすべてのエンティティ参照は、ＵＲＩ（Universal Resource Identifier）仕様に準拠する。http://www.w3.orgのＷ３ＣからＵＲＩ仕様を参照されたい。

【００８８】（ii）用語

ＣＭＬ文書を記述するのに使用される用語を、この仕様の本文中に定義する。以下のリスト中に定義する用語は、これらの定義を構築し、ＣＭＬ「プロセッサ」の動作を記述する際に使用される。ＣＭＬプロセッサは一般に、ＣＭＬコードおよび関連するアプリケーションを実行するように構成された処理装置を指す。用語は以下の通りである。することができる、してもよい（may）　適合したＣＭＬ文書およびプロセッサは、記述されたように挙動することができるが、その必要はない。しなければならない（must）　適合したＣＭＬ文書およびプロセッサは、記述されたように挙動する必要がある。そうでない場合は、以下に定義するエラーにある。エラー（error）　本仕様の規則の違反。結果は不定である。適合したソフトウェアがエラーを検出して報告することができ、それから復旧することができる。致命的誤り（fatal error）　適合したＣＭＬプロセッサが検出してアプリケーションに報告しなければならないエラー。

【００８９】Ｄ．名前空間

このセクションでは、ＣＭＬインスタンスのすべてのセクション内の名前空間の使用を詳述する。本仕様中に定義するすべての要素および属性は暗黙的に名前空間cml中にあることに留意されたい。すなわち、一般ＸＭＬ文書内で発生するＣＭＬインスタンス中の要素名messageは、ＸＭＬプロセッサにはcml:messageとして見え、ＣＭＬ属性node_idは、ＸＭＬプロセッサにはcml:node_idとして見えることになる。このセクションの後続の段落で、ＣＭＬクライアント間の名前衝突を避けるために名前空間cmlがさらにどのように細分されるかを定義する。

【００９０】「限定されない」名前空間によって導入されたすべての名前空間、例えばvxmlは、暗黙的に名前空間com.ibm.cml.vxml中にある。より一般には、ベンダ特定の名前空間は、ベンダのドメイン名から構成されるベンダ接頭部を使用する。これは、Javaのようなシステムによって使用される方式に類似する。

【００９１】ＣＭＬはまた、名前空間を使用して、異なる個々のインフォウェアからのフィールド名および値が共存できるようにする。したがって、アプリケーションcafe中のフィールドdrinkの完全に限定された名前は、cafe.drinkである。この飲物の例に対する例示的な適用は以下に述べる。ＣＭＬ中のすべてのフィールド名は常に完全に限定されることに留意されたい。すなわち、フィールド名内には、関連するジェスチャが発生するネスティング・レベルに基づく暗黙的な階層はない。

【００９２】Ｅ．ＣＭＬ属性

ＣＭＬインスタンスは、次のＸＭＬ属性を有することが

できる。別の方法で述べていない限り、すべての属性はオプションである。
（ｉ）node_id　このＣＭＬノードに対する固有の識別子。属性node_idが要求される。
（ii）title　ＣＭＬインスタンスにタイトルを指定する、人間が読めるメタデータ・ストリング。
（iii）name　ＣＭＬインスタンス内でインスタンス化されたすべてのフィールド値に対して名前空間を確立するのに使用される名前。
（iv）action　ＣＭＬのターゲット・アクションであるＵＲＬ（Uniform Resource Locator）を指定する。
（ｖ）style　関連するＸＳＬスタイル・シートのＵＲＩ。指定がない限り、ＣＭＬインタープリタは、モーダル性独立のＣＭＬインスタンスをモーダル性特定のコード化に変換するために、デフォルトで一般スタイル・シートに設定する。属性styleは、ＣＭＬ作成者がシステム全体のスタイル規則をオーバーライドまたは特殊化することを可能にする。
【００９３】Ｆ．ＣＭＬコンポーネント
ＣＭＬインスタンスは、「会話型ジェスチャ」を表す。前述のように、会話型ジェスチャは、ダイアログの基本的な構成単位であり、対話論理をモーダル性独立の形でカプセル化する。複雑な会話型コンポーネント（ダイアログ・コンポーネントまたはダイアログ・モジュールとも呼ばれる）は、後続のサブセクションで述べるより基本的な会話型ジェスチャを統合することによって構築される。これらの複雑な会話型コンポーネントは通常、例えば電話番号を得る、住所を得るなどのタスク指向である。基本的な会話型ジェスチャのＣＭＬ記述は、所望のレベルの複雑性までネストすることができる。ネスティングの他にも、複雑な会話型コンポーネントは、基本的な会話型ジェスチャを並列で、または逐次的に、あるいはその両方で結合することによっても得られる。また、複雑な会話型コンポーネントは、命令型ジェスチャ、例えばＣＦＣ（Conversational Foundation Class）を結合することによっても得られ、これは以下に述べる。あらゆるＣＭＬジェスチャがＸＭＬ要素であるものの、その逆は真ではないことにも留意されたい。すなわち、本仕様中に定義するあらゆるＸＭＬ要素がＣＭＬジェスチャであるというわけではない。多くのＣＭＬジェスチャは、サブ要素を使用して所与のジェスチャの基礎構造をカプセル化する。後続のセクションでは、「ジェスチャ」であるＣＭＬ要素は、ジェスチャ・メッセージというタイトルのサブセクション中のようにマークされる。
【００９４】ＣＭＬは、新たに出現する他のＷ３Ｃ規格、例えばＸＨＴＭＬ（Extensible HyperText Markup Language）などと相互動作するように設計される。したがってＣＭＬ要素は、ＨＴＭＬ、ＭＡＴＨＭＬなどのような他のマークアップ言語から要素を再創作するのではなく、適切なところで再利用する。このような要素は、ＣＭＬインスタンス中に埋め込まれると、例えばhtml:emなどに完全に限定される。以下の第１のサブセクションは、様々なＣＭＬ構成単位の共通面を紹介し、後続のサブセクションは、各構成単位を詳細に記述する。各ＣＭＬ基本要素が基本的な会話型ジェスチャを取り込み、ＸＭＬ属性がより特殊化された挙動をコード化するのに使用されることに留意されたい。したがって、例えば、イエス／ノーの質問はＣＭＬ基本要素であり、ユーザの確認を必要とするイエス／ノーの質問は、この基本要素を改良したものである。
【００９５】ＣＭＬはジェスチャの上書きおよびジェスチャの拡張を可能にするので、ＣＭＬの特定の実施形態中に提供されるＣＭＬジェスチャの基本セットがどんなものであるかは問題ではない。本明細書で提供されるセットおよび規則は、どんなレガシー・ページおよび対話の実施も可能にする。
【００９６】ＣＭＬジェスチャは、以下の共通ＸＭＬ属性を共用する。
action　ジェスチャの完了時に行われるアクション。属性actionは、link、return、またはsubmitのうちの１つとすることができる。
【００９７】（ｉ）ジェスチャMessage
会話型ジェスチャmessageは、ユーザに情報メッセージを伝達するのに使用される。ジェスチャmessageは通常、表示される文字列または話されるプロンプトとしてレンダリングされる。話されるメッセージの一部は、ＣＭＬインタープリタによってホスティングされている様々な個々のインフォウェアの現状態の関数とすることができる（アクセス環境の状態に関するセクション参照）。例：<message node_id="1">Your <html:em> checking</html:em> account balance is<value name="banking.checking.balance"/>after transfering<value name="banking.checking.transfer"/>to your<value name="banking.creditCard.account ' '/></message>空の要素valueは、現在の環境からの可変の情報を挿入するのに使用され、アクセス環境の状態に関するセクションで正式に定義する。
【００９８】（ii）ジェスチャHelp
会話型ジェスチャhelpは、ダイアログにトラブルが起こった場合に表示すべきコンテキスト・ヘルプをカプセル化するのに使用される。ジェスチャhelpは通常、表示されるストリングまたは話されるプロンプトとしてレンダリングされる。メッセージの一部は、ＣＭＬインタープリタによってホスティングされている様々な個々のインフォウェアの現状態の関数とすることができる。
例：
help node_id"1"You can check your account balances by specifying aparticular account. </help>
【００９９】（iii）Final
ＣＭＬ要素finalは、カプセル化されたジェスチャが首

尾よく完了したときに行われるアクション、例えばユーザ対話に基づいてカプセル化環境を更新することをカプセル化するためにジェスチャ内で使用される。

【０１００】（iv）ジェスチャBoolean：イエス／ノーの質問

会話型ジェスチャbooleanは、典型的なイエス／ノーの質問をカプセル化する。ジェスチャbooleanは、messageとして使用すべきプロンプトならびにもしあればデフォルトの応答をカプセル化する。属性require_confirmation、require_confirmation_if_no、およびrequire_confirmation_if_yes（デフォルトではすべてfalse）によって、インフォウェア・アプリケーションがダイアログを改良することができる。

例：

```
<boolean default="y"
node_id="1"
require_confirm_if_no="true">
<grammar type="text/jsgf">
(yes | yeah) {yes} | (no nay) {no}
</grammar>
<message>
Please confirm that you would like to stay at the
<value href="travelCenter.hotel.selected"/>
</message>
</boolean>
```

【０１０１】（ｖ）ジェスチャSelect

会話型ジェスチャselectは、ユーザが一組の選択肢から選び取ることを期待される場合のダイアログをカプセル化するのに使用される。これは、プロンプト、デフォルト選択、ならびに合法な選択肢のセットをカプセル化する。要素selectの属性は、相互排他的選択を達成する（ラジオ・ボタンのグループとして視覚的にレンダリングされる）、範囲から選択する、スクロールバーとして視覚的にレンダリングするなどのためにジェスチャを改良する。selectのサブ要素は、以下のものを含む。

choices

値と参照のいずれかによって埋め込まれた可能な選択肢のリストを含む。要素choicesは、１つまたは複数のchoice要素のリスト、ならびにもしあればデフォルト選択を指定するせいぜい１つのdefault要素を含む。

predicate

選択が満たすべきテストをカプセル化する述語。

help

ダイアログが行き詰まった場合に提供すべきヘルプ。

error

predicateが失敗の場合に使用すべきメッセージを含む。

例：

```
<select name="portfolio.fund"
node_id="1"
require predicate= "yes">
<message node_id="2">
Which of your positions would you like to check?
</message>
<help>
You can specify the names of stocks or funds you o
wnand we will report your current position.
</help>
<choices>
<var name="possibleChoices"/>
<default value="possibleChoices">
Check the position of all holdings</default>
</choices>
<predicate>
<conduction>
fund in possibleChoices
</condition>
<error>
Sorry, you do not appear to own any shares in
<var name="portfolio.fund"/>
</error>
</predicate>
</select>
```

【０１０２】（vi）Predicate

要素predicateは、特定の会話型ジェスチャの結果を妥当性検査するための規則をカプセル化するためにＣＭＬ中で使用される。テスト述語は、その開示を参照により本明細書に組み込むＷ３Ｃからのxpath仕様すなわちXML Path Language, W3C Proposed Recommendation中に定義される表現構文および意味論を使用して単純な条件文として表される。http://www.w3.org/tr/xpathを参照されたい。xpathは、文書ツリーの異なる部分にアクセスするための表現構文を指定し、アプリケーション・バックエンドへの呼出しを必要とする妥当性検査は、個別に扱われる。

【０１０３】predicate要素を含む会話型ジェスチャは、テストが失敗した場合に適切な属性を介して行うべきアクションを限定する。

【０１０４】（vii）Grammer

ＣＭＬサブ要素grammarは、VoiceＸＭＬ中で要素grammarの後にモデリングされる。サブ要素grammarは、文法フラグメントをコード化し、サブ要素helpが、どの発話が許容されるかをユーザに示すために再生される適切なヘルプ・メッセージをカプセル化する。適切な場合に、ＣＭＬジェスチャは、ＣＭＬインタープリタによってより複雑な文法に組み立てられる文法フラグメントを提供することができる。

【０１０５】サブ要素grammarは、入力、特にスピーチを処理する規則として一般化することができる。これらの規則は、厳密にすることができる、または、処理に使

用すべき遠隔リソース（ＵＲＬ）を記述して、どのデータ・ファイルでどの処理を行い、どのアドレスにどんな結果を返さなければならないかを記述する引数を、これらのリソースに渡すために提供することができる。一般に文法は、適正範囲内に定義するかまたはＵＲＬを介して定義することができる。

【０１０６】さらに、この処理を、オブジェクト・タグ、例えば<object>...<object>によって宣言することも可能である。オブジェクト・タグによって、ＣＦＣ（Conversational Foundation Classes）またはＣＡＰ（Conversational Application Platform）サービスをロードすることができる（例えば、上に参照したＵＳ９９／２２９２７（整理番号ＹＯ９９９－１１１）として識別されるＰＣＴ国際特許出願を参照されたい。ここでＣＡＰはＣＶＭすなわち会話型仮想マシン（Conversational Virtual Machine）に相当する。）引数は、ＸＭＬ属性および変数を使用してオブジェクトに渡すことができる。結果は、同様の変数プレースホルダを介して返すことができる。これにより、これらのオブジェクト呼出しは、環境にアクセスしてそれを修正することができる。

【０１０７】オブジェクトは、以下の値をとることのできる属性executeによって限定することができる。すなわち、parallel（並列でブロックされずに実行され、環境への進行中の影響を、その完了前の実行中に知らせることができる）、asynchronous（非同期でブロックされずに実行され、環境の更新が完了したときにイベントを介して知らせる）、blocking（環境の更新および継続の前にブラウザがオブジェクト呼出しの完了を待機する）である。

【０１０８】処理を分散させるのに必要なすべての情報は、上で参照したＵＳ９９／２２９２５（整理番号ＹＯ９９９－１１３）として識別されるＰＣＴ国際特許出願に記載されており、これは、会話型アプリケーションの分散を可能にする体系およびプロトコルを定義する。したがって、この国際特許出願は、そのような分散をどのように行うことができ、それがどのようにこの場合のクライアント・ブラウザとサーバ・ブラウザとの間、ならびにローカル・エンジンとサーバ・エンジンとの間に処理を分散させることを可能にするかを記述する。これにより、ネットワークにわたる入力／出力イベントの処理の分散が可能である。

【０１０９】（viii）ジェスチャMenu

ジェスチャmenuは、ジェスチャselectの特殊なケースである。ジェスチャmenuは、ユーザがあるアプリケーションの異なるサブパートをナビゲートするのを助けるダイアログをカプセル化するのに使用される。ジェスチャselectを使用して、同じ結果を達成することができる。しかし、明示的なmenuジェスチャを有することにより、作成者は、selectジェスチャが使用されている理由に関してより意味的な情報を提供することができる。以下の例で、要素menuは、属性actionがlinkに設定されている要素selectに相当することに留意されたい。

例：

```
<menu name="main">
<choice value="#query">Ask a question</choice>
<choice value="#browse">Browse available categories</choice>
</menu>
```

各選択における属性valueの値は、その選択に対するＵＲＩターゲットを指定する。

【０１１０】（ix）ジェスチャUser Identification

会話型ジェスチャuser_identificationは、ユーザ・ログインおよび認証をカプセル化するのに使用される。これは総称的なものとされ、スタイル規則を介して特定のユーザ対話環境用に特殊化される。

【０１１１】サブ要素userおよびidentifyは、ユーザ名および認証の情報を得るための会話型ジェスチャをカプセル化する。要素predicateは、ユーザが首尾よく認証されたかどうかを確認するためのテストを提供する。

例：

```
<user_identify name="login"
require_predicate="yes"
on_fail="retry"
node_id="2">
<message node_id="3">
To use this service, you first need to login using
 your nameand personal identification.
</message>
<user name="userid"
node_id="4">
what is your user id?
</user>
<identify name="pin"
node_id="4">
Please provide your user authentication.
</dentify>
<predicate>
<conditon>
backend.authenticate(user id,pin)
</condition>
<predicate>
<error>
Sorry, login for <var name="userid"/>
with identification <var name="pin"/> failed.
</error>
</user_identify>
```

このジェスチャに対する変形は、例えば、識別ジェスチャ（例えばその人が誰かを識別する）、検証ジェスチャ（例えば要求者の認証）、スピーチ生物測定法（例えば

米国特許第５８９７６１６号）の間の明示的な区別に有用となることができる。

【０１１２】（ｘ）ジェスチャ制約付きのInput

ＣＭＬは、日付や通貨などのユーザ入力を得るための、いくつかの定義済みダイアログ・コンポーネントを提供する。通常、このような入力は、これまでに列挙した様々な選択ジェスチャよりも拡張可能であり、単純な編集フィールドを介して従来の視覚インタフェース中で実現される。しかし、このような入力ジェスチャに対するドメイン特定の制約をカプセル化することは、口頭の対話を構築する際に有利である。また、このようなドメイン特定の制約は、通常、サーバにサブミットする前にユーザ入力の妥当性検査を行うＨＴＭＬページ内のクライアント側スクリプトとして、今日のWWWインタフェース中で実施されていることにも留意されたい。ＣＭＬでは、今日のWWW上で広く使用されているこれらの入力ジェスチャを、標準的なユーザ・レベルのタスクを行うために形式化する。ＣＭＬはまた、この基本的な入力ジェスチャのセットを時の経過に伴って拡張することのできる拡張機構も提供する。このリスト中に定義するすべてのＣＭＬ要素はジェスチャであることに留意されたい。

（１）Date　日付を指定する。

（２）Time　時間を指定する。

（３）Currency　通貨額を指定する。

（４）Credit Card　クレジット・カード（カード・タイプ、カード番号、有効期限を含む）を指定する。

（５）Phone　電話番号を指定する。

（６）Email　電子メール・アドレスを指定する。

（７）URL　ＵＲＬを指定する。

（８）Snail Address　郵便番号、国／州／市、通りを含めた「スネイル・メール」アドレスを指定する。

【０１１３】制約付きの入力ジェスチャは、文法を他の入力フィールドに渡すことによって容易に拡張することができる。このジェスチャはさらに、ローカライズできる（すなわち、国際化でき、地域的な趣を持たせることのできる）トランスコーディング規則に関連付けることもできることに留意されたい。これは、すべてのジェスチャおよびジェスチャ・ベースのトランスコーディング規則に実際に拡張可能なステートメントである。ロケーション（すなわち電話番号、ＩＰアドレスの起点、ユーザに関して分かっている（ユーザのローカル装置／ブラウザ上にあるかまたはクッキーを介してサーバに送信された）プリファレンス））に基づいて、ジェスチャを別の言語で表すことができ（すなわち、「Select yes or no」が「Selectionnez oui ou non」になるなど）、あるいは地理に適合させることができる（例えば、zipコードが郵便番号になる）。

【０１１４】（xi）ジェスチャ制約のないInput

会話型ジェスチャinputは、入力制約がより複雑な（またはおそらく存在しない）場合の、得られたユーザ入力に使用される。このジェスチャは、ユーザ・プロンプト、要求されている情報のアイテムに関するアプリケーション・レベルの意味構造、およびもしかすると入力の妥当性をテストするためのpredicateをカプセル化する。アプリケーション特定の意味制約を伴うジェスチャinputは、前のセクションで考察した、組み込まれた制約付き入力のジェスチャのセットを拡張する手段を提供することに留意されたい。

例：

```
<Input node_id="1">
<Message>...</Message>
</Input>
```

【０１１５】（xii）ジェスチャSubmit

会話型ジェスチャsubmitは、パッケージされて、含んでいるＣＭＬインスタンスから返されるべき、環境からのコンポーネントを指定する。これはまた、使用すべきプロンプト、ならびにカプセル化された環境の状態をサブミットすべきターゲットＵＲＩもカプセル化する。

例：

```
<submit target="uri">
<env name="location.state"/>
<env name="location.city"/>
</submit>
```

サブ要素envは、囲んでいるジェスチャがサブミットすべき環境のコンポーネントを指定する。

【０１１６】様々なＣＭＬ属性およびコンポーネントを以上に説明したが、ＣＭＬのこの実施形態の他の態様を説明しながら、その他の属性およびコンポーネントを以下に呈示および定義することを理解されたい。その他の属性およびコンポーネントは、本発明の教示に従って定義される場合があることを理解されたい。すなわち、本発明は、この詳細な説明で述べる特定の属性およびコンポーネントに限定されるものではない。

【０１１７】Ｇ．結合イベント

ＣＭＬは、アプリケーション作成者が「論理入力イベント」、ならびに、そのような論理イベントと、定義された論理イベントをトリガする実際の「物理入力イベント」との間の関連を定義するための、フレキシブルかつ拡張可能な機構を提供する。ＣＭＬジェスチャは、定義された論理イベントが受け取られるとき、ＣＭＬジェスチャが扱う準備のできている論理イベントを、ＣＭＬ属性triggerを介して宣言し、そのトリガ・リスト中にマッチング・イベントを有する最も近くの囲んでいるジェスチャが、そのイベントを扱う。ＣＭＬ属性triggerは、ジェスチャがそれに論理的に結合されたイベントによってトリガされるようにする。この機構は、例によって最もよく示される。以下に示すＣＭＬコードの断片では、アプリケーションは、helpを論理入力イベントとして定義し、これを異なる２つのモーダル性にある物理イベントに結合し、最終的に、helpイベントを扱うＣＭＬ

ジェスチャを宣言する。

```
例：
〈cml name="travel"〉
〈bind-event      logical="help"
                  modality="dtmf"
                  physical="*"/〉
〈bind-event logical="help"
                  modality="qwerty"
                  physical="h"/〉
〈help name="help"
      trigger="help"〉
 Top-level application help
〈/help〉
 ...

〈/cml〉
```

ＣＭＬ要素bind-eventは、３つの属性をとる。

（１）logical　定義されている論理イベントの名前を指定する。

（２）modality　イベントが結合されている対話モーダル性を指定する。

（３）physical　論理イベントに結合させる物理イベントを指定する。

【０１１８】アプリケーションを構成するＣＭＬジェスチャによって扱われない入力イベントは、helpなどの標準的なプラットフォーム・イベントがデフォルト・ハンドラによって扱われるＣＭＬインタープリタにバブルアップする。バブルアップとは、トリガ値と一致するジェスチャを探索することが、一致するジェスチャがなくなるまで階層的に、最も近くの囲んでいるジェスチャからより高いジェスチャにバブルアップするという意味である。このような場合、トリガは、ブラウザから、さもなければ基礎をなすプラットフォーム（例えばＹＯ９９９－１１１の会話型仮想マシン）から提供されるサービスに関連付けるべきである。どれも満たされない場合、そのイベントは無視されるか、または入力が理解されず（またはサポートされず）無視されたことを説明するデフォルト・メッセージがユーザに返される。しかしこれらは、ブラウザおよび基礎をなすプラットフォームを実施する選択肢であって、言語の選択肢ではない。機構bind-eventは、プラットフォームの挙動をオーバーライドするように設計され、これは、ユーザ入力をＣＭＬジェスチャにマッピングするための排他的機構として使用されるものではないことに留意されたい。したがって、アプリケーション中の有効なすべての口頭発話を適切なジェスチャに結合するのに要素bind-eventを使用することは反対される。

【０１１９】さらに、要素bind-event中の属性modalityを省略することにより、指定の物理結合がすべてのモーダル性で関連付けられることにも留意されたい。要素bind-event中の属性physicalの値を省略することは、非結合の、すなわち物理イベントに結合されない論理イベントを宣言する。

【０１２０】Ｈ．グループ化ジェスチャおよび定義フォーカス

会話型ジェスチャは、特定のユーザ・インタフェースを実現するために特定のモーダル性にされるとき、適切にグループ化されて、ユーザがインタフェースの関係する部分と対話することを可能にする。この断定を理解するために、ユーザ対話をいくつかのＨＴＭＬページにわたって分割し、インタフェースの関係する部分が同じページ上に表示されるＷＷＷアプリケーションを考察されたい。同様に、スピーチ・インタフェースは、ユーザがいくつかの関係するコマンドのいずれかを所与の時間に指定することを可能にする。

【０１２１】こうした形のジェスチャのグループ化は、アプリケーションがオーサリングされているときにもっともよく取り込まれる。このようなグループ化は、モーダル性独立の場合もあり、そうでない場合もある。ＣＭＬによれば、アプリケーション作成者は、両方の形のグループ化をカプセル化することができる。

【０１２２】会話型ジェスチャは、ＣＭＬ要素groupを使用してグループ化される。要素groupはさらに、属性id、modality、classによって限定される。属性idは、ジェスチャをグループ化するのに最低限必要である。属性modalityは、もしあれば、指定のグループ化をモーダル性特定にすることを宣言する。属性classは、関係する要素をさらに選択してＣＭＬをＨＴＭＬのような言語にトランスコーディングするために、ＨＴＭＬのclass属性と同様の方式で使用することができる。

【０１２３】デフォルトで、単一のgroup要素に囲まれたＣＭＬジェスチャは、含まれるいずれかのジェスチャとユーザが対話できるようにするユーザ・インタフェースにマッピングする。ＨＴＭＬの場合、これは、単一ページにトランスコーディングされているジェスチャとなり、VoiceＸＭＬの場合、これは、並列で活動化されている対応するフォームとなる。

【０１２４】ジェスチャのグループを並列で活動化することは、混合イニシアチブＮＬインタフェースを実施する方式であり、所与の時間にサポートされる各コマンド／照会は、ジェスチャの中から構築されたフォーム（すなわちジェスチャのグループはフォームと呼ばれる）を特徴とすることに留意されたい。入力／出力イベントが発生するとき、ブラウザまたは基礎をなすプラットフォームから提供されるダイアログ・マネージャは、活動状態にされた異なるフォーム中のジェスチャが何であるかを推測することになり、それらの関連する属性（ジェスチャに関連付けられた環境変数）を限定できるようにする。フォームのすべての必須属性が値を受け取ると、アクションは一義化されて実行されると見なされる。以下

に述べるが、ＸＦＯＲＭＳを使用して属性間の追加の制約を表すこともできることに留意されたい。並列活動化に関する考察に対する、上で参照した整理番号ＹＯ９９８－３９２によって識別される特許出願と、K.A.Papineni他の「Free-flow dialogmanagement using forms」, Proc. Eurospeech,1999、およびその開示を参照により本明細書に組み込むK.Davies他の「The conversational telephony systemfor financial applications」,Proc. Eurospeech,1999も参照されたい。

【０１２５】要素groupのインスタンスは、内部グループ要素が、囲んでいる要素中に指定されたものとは異なる値を属性modalityまたはclassに対して指定しない限り、ネストすることができない。

【０１２６】ＸＦＯＲＭＳ（http://www.w3.org/MarkUp/Forms/）のような努力は、同じバックエンドを保持しながら異なる種類のブラウザに向けた表示の置換を容易にする試みにおいて（ただしＸＦＯＲＭＳは異なるモーダル性に対処するのに失敗している）、図４に示すようにフォームを３層（プレゼンテーション、論理、データ）に分割することにより、既存のマークアップ言語に関連する問題を解決しようとしてきた。ＸＦＯＲＭＳデータ層は、アプリケーション開発者がフォームに対してデータ・モデルを定義することを可能にする。開発者は、組込みデータ・モデルを使用することもでき、自分自身のものをロールすることもできる。ＸＦＯＲＭＳは、ＸＭＬスキーマ上で行われている作業の最上部にデータ・モデルを構築している。論理層は、アプリケーション開発者が、例えば現総計に対する、またはあるフィールドがファイルされるのに別のフィールドを必要とする場合の、フィールド間の依存性を定義することを可能にする。ＸＦＯＲＭＳは、スプレッド・シートおよび既存のフォーム・パッケージへの広く行き渡った親しみの上に構築する軽量表現構文をサポートする。アプリケーション開発者は、追加のフレキシビリティが必要なときに、依然としてスクリプトを呼び出すことができる。プレゼンテーション層は、フォーム制御用のマークアップおよび他のＨＴＭＬマークアップからなり、各制御は、データ・モデル中のフィールドに結合される。「getter」および「setter」関数は、正規の表現を内的に保持しながら、プレゼンテーションが例えば日付および通貨に対するユーザのプリファレンスに一致するようにし、したがって、フォーム処理を簡略化する。同じデータ・フィールドが、２つ以上のプレゼンテーション制御を結合されることができる。いずれかの制御で値が変更されると、次いで、他のすべてが自動的に更新される。

【０１２７】本明細書で説明したように、ＸＦＯＲＭＳは、プレゼンテーションからデータを分離するためのバックエンド機構を提供する。ＣＭＬは、論理およびプレゼンテーションの部分をさらにプレゼンテーション・レンダリング（すなわち、対話情報のないモーダル性独立のレンダリング）／対話（可能性あるモーダル性依存の表面的入力を足したもの）／コンテンツ（すなわち、バックエンド・データと論理情報を足したものから対話に関係するすべての論理対話を引いたもの）に分離するための機構を提供する。この創意に富んだ概念を図５に示す。前に説明したように、本発明のプログラミング・パラダイムは、プレゼンテーション／モーダル性特定のレンダリングＡと、対話Ｂと、コンテンツおよびバックエンド／アプリケーション論理Ｃとを分離する。図５はまた、前述のＸＦＯＲＭＳのバックエンド機構も示し、データＤはバックエンドＥから分離している。図６は、どのようにフォーム・ベースの混合イニシアチブＮＬＵ（自然言語理解、natural language understanding）アプリケーションがＣＭＬで書かれるかを表している。文字ＡおよびＣは、図５にあるのと同じアイテムを示す。Ｂ'で示したブロックでは、対話／ダイアログ情報がＣＭＬで記述される。この部分は、活動化可能な各トランザクションを実現するために発生する必要のある対話（必須およびオプション）を記述する。このためには、基礎をなす属性データ構造に関連する制約およびデータ・モデルを取り込むＸＦＯＲＭＳコンポーネントを追加する。エンジン制御および表面変更の部分は、会話型エンジン、特にダイアログ・マネージャおよびＮＬＵエンジンの挙動を最適化するのに使用される追加の制御パラメータを取り込む。ＣＭＬの部分は、前に述べたように他のモーダル性でレンダリングするのに使用できることに留意されたい。ブロックＦは、ブロックＢ'に従って利用できる例示的なフォーム（例えばミューチュアル・ファンド・デモ・フォーム）を示す。

【０１２８】Ｉ．データ・モデルおよびデータ・モデル
ＣＭＬは、ＸＭＬスキーマおよびＸＭＬフォームに関するＷ３Ｃワークの結果によって指定されるデータ・モデルまたはデータ・モデル基本要素を定義する。http://www.w3.orgを参照されたい。

【０１２９】Ｊ．アクセス環境
ＣＭＬジェスチャは、集合的に「環境」と呼ばれる変数の集合を定義する。ＣＭＬ文書が横断されるとき、環境中の変数は、成功したユーザ対話から得られる値に束縛される。環境は、以下に述べるが、要素var、value、assignを介してＣＭＬジェスチャ内でアクセスし操作することができる。このような名前はすべて、常に完全に限定されることに留意されたい。

【０１３０】（ｉ）var　要素varは、現行の環境で、変数を宣言し、（かつ任意選択で）初期設定する（それを初期値に割り当てる）。属性nameは、変数の名前を指定する。初期値は、要素assignに対して指定するのと同じ構文を使用して指定することができる。以下を参照されたい。

【０１３１】（ii）assign　要素assingは、環境中にすでに存在する変数に値を割り当てる。すなわち、要素as

signは、環境中の値を束縛するのに使用される。属性nameは、束縛すべき変数を指定する。束縛すべき値は、属性exprの値として、xpathによって使用されるのと同じ表現構文を使用して指定することもでき、あるいは、割り当てるべき値は、要素assignのコンテンツとして指定することもできる。要素assignは通常、直接的なユーザ対話によって設定されない中間変数を束縛または更新するのに使用される。

【０１３２】（iii）value　要素valueは、定義された変数の値を取り出す。すなわち、空の要素valueの属性nameは、その値を環境中で検索すべき変数を指定する。属性nameの値は、部分的にまたは完全に限定された名前であり（名前空間に関する前述のセクション参照）、含んでいるＣＭＬジェスチャのコンテキストで解釈される。

【０１３３】上に定義したように、変数は、それらを割り当てることができる前に宣言されなければならないことに留意されたい。

【０１３４】Ｋ．ＣＭＬ横断モデル

ＣＭＬでオーサリングされたインフォウェアは、複数のユーザ・エージェント間を調停する会話型シェルによってホスティングされ、この会話型シェルを、以後、ＣＭＬインタープリタと呼ぶ。横断モデルは、図１３および１４のコンテキストでさらに考察し、例示することを理解されたい。ユーザ対話は、ＣＭＬインタープリタがＣＭＬインスタンスをＨＴＭＬやVoiceＸＭＬなどの適切なモーダル性特定の言語にマッピングすることによって進行する。これらのモーダル性特定の表現は、モーダル性特定バージョンのダイアログをレンダリングする適切なユーザ・エージェントに渡される。

【０１３５】ＣＭＬからモーダル性特定の表現への変換は、ＸＳＬ変換規則（ＸＳＬＴ）に支配されることが好ましい。他の変換機構も使用できることに留意されたい。ＸＳＬＴは、好ましい実施形態に提案される一手段にすぎない。例えば、ＪＳＰ、Java Server PagesまたはJava Beans、ならびに規則に基づいてジェスチャをそれらのターゲット・レンダリングに変換する他の技術を使用することもできる。このような実施の一例は、各ジェスチャにジャバ・ビーンを関連付けるものである。ジャバ・ビーンは、各モーダル性でそれ自体のレンダリングを（ＪＳＰを介して）伝える。したがって、本発明はＸＳＬＴに限定されない。いずれの場合にも、これらのＸＳＬ規則は、モーダル性特定である。ＣＭＬインスタンスを適切なモーダル性特定の表現にマッピングするプロセスで、ＸＳＬ規則は、モーダル性特定のユーザ対話の実現に必要な情報を追加する。例として、要素selectをVoiceＸＭＬに変換するとき、関係するＸＳＬ変換規則は、その会話型ジェスチャに有効な選択肢をカバーする文法の生成を扱う。

【０１３６】ＣＭＬインスタンスをＨＴＭＬなどのモーダル性特定の表現に変換するプロセスは、単一のＣＭＬノードを出力表現中のノードの集合にマッピングすることになる可能性がある。これらの様々な表現にわたる同期化を助けるために、ＣＭＬ属性node_idが、所与のＣＭＬノードから得られたすべての出力ノードに適用される。所与のＣＭＬインスタンスが、適切なモーダル性特定のＸＳＬ規則によって異なる表現、例えばＨＴＭＬおよびVoiceＸＭＬにマッピングされるとき、出力におけるツリーの形は、様々なモーダル性にわたっておそらく非常に異なる。しかし、属性node_idは、モーダル性特定の各表現から発生源であるＣＭＬノードへの概念上のバックリンクを提供することによって、これらの表現の間で同期をとることを可能にする。上で参照した米国第６０／１２８０８１号（整理番号ＹＯ９９９－１７８）として識別される米国仮特許出願に、マルチモーダル・アプリケーションをしっかりとサポートできるプラットフォーム（マルチモーダル・シェル）をどのように開発するかに関する記述が提供されている。この機構は、次のように動作する。各モーダル性は、それがサポートするコマンドと、それらの実行が他の登録済みモーダル性に与えることになる影響とをマルチモーダル・シェルに登録する。明らかに、ここでの場合、ＣＭＬページを解析しジェスチャをトランスコーディングするとき、各ジェスチャは、マルチモーダル・シェル中のデータ構造（すなわちテーブル）中に保持される。所与のモーダル性のＩ／Ｏイベント時に、活動化されたジェスチャを見つけるためにnode_id情報が使用され、テーブル（すなわちＣＭＬ文書ダイアログ・ツリー）から、活動化されたモーダル性ならびに他のモーダル性への影響が即座に見つかる（すなわち、各ビューを更新するかまたはＣＭＬサーバ上の新しいページを取り出す）。

【０１３７】ユーザ対話が進行するにつれ、現行のＣＭＬインスタンスによって環境中で定義された変数は、妥当性検査された値に束縛される。まず、登録されたモーダル性特定のユーザ・エージェントの１つでこの束縛が発生する。登録されたユーザ・エージェントは、更新された環境および完了したばかりのジェスチャのnode_idからなる適切なメッセージを会話型シェルに送る。更新された束縛がＣＭＬインタープリタに伝えられると、ＣＭＬインタープリタは、完了したばかりのジェスチャのnode_idで、登録されたすべてのユーザ・エージェントにメッセージを送る。登録されたユーザ・エージェントは、このメッセージを受け取ると、まずそれらのプレゼンテーションに影響する環境の部分に対してＣＭＬインタープリタに照会することによって、それらのプレゼンテーションを更新する。

【０１３８】Ｌ．特定ユーザ・インタフェース言語へのＣＭＬの変換

ＣＭＬは、ＸＳＬで表現される変換規則によって、ユーザ・インタフェース（ｕｉ）特定のコード化、例えばＨ

ＴＭＬに変換される。このセクションは、ＸＳＬ変換についてのいくつかの背景材料で始まり、次いで、本発明によってＸＳＬがどのようにＣＭＬおよびマルチモーダル・ブラウザのコンテキストで使用されるかについての例を呈示する。

【０１３９】（ｉ）ＸＳＬ変換の背景情報

Ｗ３Ｃ　ＸＳＬ変換（xslt）仕様は、James Clark編Proposed Recommendation: XSL Transformations (xslt) Version 1.0, reference: W3C Proposed Recommendation 8-October-1999としてリリースされており、この開示を参照により本明細書に組み込む。上に参照したW3C Proposed Recommendationは、Ｗ３Ｃ Style活動の一部である。具体的には、xslt仕様は、ＸＭＬ文書を他のＸＭＬ文書にトランスコーディングするための言語であるxsltの構文および意味論を定義する。xsltは、ＸＭＬに対するスタイル・シート言語であるＸＳＬの一部として使用するために設計されている。xslt言語中の変換は、xsltによって定義される要素とxsltによって定義されない要素とを含むことのできるＸＭＬ Recommendation中の名前空間に準拠した、うまく形成されたＸＭＬ文書として表現される。xsltで表される変換は、ソース・ツリーを結果ツリーに変換するための規則を記述する。変換は、パターンをテンプレートと関連付けることによって達成される。パターンは、ソース・ツリー中の要素と突き合わせられる。テンプレートは、結果ツリーの一部を生み出すためにインスタンス化される。結果ツリーは、ソース・ツリーとは別個である。結果ツリーの構造は、ソース・ツリーの構造とは全く異なるものとすることができる。結果ツリーを構成する際、ソース・ツリーからの要素は、フィルタリングしてリオーダーすることができ、任意の構成を追加することができる。xsltで表された変換は、スタイル・シートと呼ばれる。xslt仕様は、ＸＭＬとＨＴＭＬの両方のフォーマットで入手可能である。

【０１４０】（ii）ＸＳＬ変換の例

以下は、ＣＭＬコード、ＸＳＬ変換規則と、それぞれの変換から得られたＨＴＭＬ、ＷＭＬ、およびVoiceＸＭＬコードを示すコード化の例である。

【０１４１】以下のコードは、ＣＭＬと、レガシーＭＬページ（それぞれＨＴＭＬ、VoiceＸＭＬ、およびＷＭＬ）を作成するのに使用されていた異なるジェスチャ・ベースのＸＳＬ規則とによって書かれたページの完全な例を示す。各ページは、後続の図で示すような特定のレンダリングに関連する。この例は、異なる情報サービスへのアクセス、すなわちニュース、ビジネス、スポーツ、旅行、天気、芸能を提供するサイトのものである。

【０１４２】（ａ）ＣＭＬコード

これは、例に関連するソースＣＭＬページを記述する。

```
<!--$Id: cnn.cml,v 1.19 2000/02/01 Exp $-->
<!--Description: CNN Mobile In cml -->
<cml name="cnn"
     node_id="1"
     title="CNN Mobile News">
  <menu name="cnn.command"
        node_id="2" >
    <choices node_id="3" >
      <default value="#cnn.query">Select News Stories</default>
      <choice value="#cnn.exit"
              require_confirmation="true">
        Exit </choice>
      <choice value="#cnn.applicationHelp">Help</choice>
    </choices>
</menu>
<cml name="cnn.applicationHelp"
     title="About CNN Mobile"
     node_id="4"
     action="return">
  <message
          node_id="5" >
    This application allows you to select and view CNN news stories
  </message>
</cml>
<cml name="cnn.exit"
     node_id="6"
```

```
    title="Exit CNN Mobile News"
    action="submit">
  <message node_id="60">
    Thankyou for using the CNN news service
  </message>
</cml>
<group node_id="7"
      groupId="query">
<cml name="cnn.query"
    title="Search CNN Mobile News"
    node_id="8" >
  <menu name="cnn.query.topic"
       node_id="11"
       title="Topic Selection">
    <choices node_id="12" >
     <choice value="#cnn.query.news"> News </choice>
    <choice value="#cnn.query.business"> Business </choice>
     <choice value="#cnn.query.sports">
       <grammar> (sport | sports" </grammar>
       Sports
     </choice>
     <choice value="#cnn.query.travel"> Travel </choice>
     <choice value="#cnn.query.weather"> Weather </choice>
     <choice value="#cnn.query.show">
       <grammar > show [business] </grammar>
       Show Business
     </choice>
    </choices>
  </menu>
 </cml>
 <cml name="cnn.query.news"
      title="News Channel"
      node_id="13"
      action="submit">
   <select name="cnn.query.part">
     <message node_id="9" >
     Which part of today's news would you like to read?</message>
   <choices
           node_id="10" >
      <choice value="h"> Headlines</choice>
      <choice value="1"> first story </choice>
<choice value="2"> second story </choice>
     <choice value="3"> third story </choice>
   </choices>
</select>
<select name="cnn.query.interest">
  <message node_id="14" >
    Which news category would you like to read?
  </message>
 <choices node_id="15" >
```

```
    <choice value="business">
      <grammar type="text/jsgf">
        business {BIZ}</grammar>
    Business
    </choice>
    <choice value="africa">
     Africa</choice>
    <choice value="world"> World </choice>
    <choice value="United states"> United states </choice>
    <choice value="europe"> Europe </choice>
    <choice value="Asia"> Asia</choice>
    <choice value="me"> Middle East</choice>
    <choice value="america"> America </choice>
   </choices>
 </select>
</cml>
<cml name="cnn.query.business"
     title="Business Channel"
     action="submit"
     node_id="16" >
  <select name="cnn.query.part">
 <message node_id="9" >
      Which part of today's news would you like to read?</message>
    <choices
            node_id="10" >
     <choice value="h"> Headlines</choice>
      <choice value="1"> first story </choice>
      <choice value="2"> second story </choice>
      <choice value="3"> third story </choice>
    </choices>
  </select>
  <select name="cnn.query.interest">
    <message node_id="17">
      Which business category would you like to read?</message>
    <choices node_id="1S">
      <choice value="NEWS"> news </choice>
      <choice value="IN"> indexes </choice>
      <choice value="CU"> exchange rates </choice>
      <choice value="MET"> metals </choice>
    </choices>
  </select>
</cml>
<cml name="cnn.query.weather"
     title="Weather Channel"
     action="submit"
     node_id="19" >
  <select name="cnn.query.part">
 <message node_id="9" >
     Which part of today's news would you like to read?</message>
   <choices
```

```
            node_id="10" >
     <choice value="h"> Headlines</choice>
      <choice value="1"> first story </choice>
      <choice value="2"> second story </choice>
      <choice value="3"> third story </choice>
    </choices>
   </select>
   <select name="cnn.query.interest">
     <message node_id="20">
      Which region are you interested in?</message>
    <choices node_id="21">
      <choice value="us"> United states </choice>
      <choice value="europe">
        <grammar type="text/jsgf"> (euro | Europe) </grammar>
        Europe
      </choice>
      <choice value="JP"> Japan </choice>
      <choice value="AU"> Australia </choice>
      <choice value="AS"> Asia </choice>
    </choices>
   </select>
</cml>
<cml name="cnn.query.travel"
     title="Travel Section" action="submit"
     node_id="522" >
   <select name="cnn.query.part">
   <message node_id="9" >
        Which part of today's news would you like to read?</message>
      <choices
              node_id="10" >
       <choice value="h"> Headlines</choice>
        <choice value="1"> first story </choice>
        <choice value="2"> second story </choice>
        <choice value="3"> third story </choice>
      </choices>
    </select>
    <select name="cnn.query.interest">
      <message node_id="23">
        Which city do you want to visit?</message>
      <choices node_id="24">
        <choice value="AMSTERDAM">AMSTERDAM</choice>
        <choice value="COPENHAGEN">COPENHAGEN</choice>
        <choice value="HELSINKI">HELSINKI</choice>
        <choice value="HONGKONG">HONGKONG</choice>
        <choice value="LONDON">LONDON</choice>
        <choice value="OSLO">OSLO</choice>
        <choice value="PRAGUE">PRAGUE</choice>
        <choice value="SINGAPORE">SINGAPORE</choice>
        <choice value="STOCKHOLM">STOCKHOLM</choice>
        <choice value="SYDNEY">SYDNEY</choice>
```

```
      </choices>
    </select>
</cml>
<cml name="cnn.query.sports"
     action="submit"
     title="Sports Channel"
     node_id="25" >
  <select name="cnn.query.part">
  <message node_id="9" >
       Which part of today's news would you like to read?</message>
     <choices
              node_id="10" >
      <choice value="h"> Headlines</choice>
       <choice value="1"> first story </choice>
       <choice value="2"> second story </choice>
       <choice value="3"> third story </choice>
     </choices>
   </select>
   <select name="cnn.query.interest">
     <message node_id="26">
       What sports are you interested in?</message>
     <choices node_id="27">
       <choice value="AS"> Asia </choice>
       <choice value="w"> world </choice>
       <choice value="eu"> europe </choice>
       <choice value="us"> united states </choice>
       <choice value="nba"> NBA </choice>
       <choice value="nkl"> nbl </choice>
       <choice value="EF"> Europoean football </choice>
  </choices>
   </select>
 </cml>
 <submit target="htUp://raman.almaden.ibm.com/cgi-bin/cnn.cgi">
   <message node_id="28">
     executing <value name="cnn.command"/>
     for <value name="cnn.query.part"/>
     stories about <value name= "cnn.query.interest"/>
     from topic <value name="cnn.query.topic"/>
   </message>
   <env name="cnn.command"/>
   <env name="cnn.query.topic"/>
   <env name="cnn.query.interest"/>
   <env name="cnn.query.part"/>
  </submit>
 </group>
 <submit target="http://raman.almaden.ibm.com/cgi-bin/cnn.cgi">
 </submit>
</cml>
```

【0143】(b)ジェスチャXSL

以下の例は、CMLページをジェスチャごとにHTML

ページにトランスコーディングするのに使用される、CMLからHTMLへのジェスチャ・ベースのXSL規則

を示す。任意の可能なＣＭＬページをトランスコーディングするのに必要なジェスチャ・ベースのトランスコーディング規則がすべてあるわけではない。これは、この方法の例示と見なすべきである。ＸＳＬ構文は、従来のＸＳＬＴ規則に従う。例えば、http://www.w3.org/1999/XSL/Transformを参照されたい。

```
!—$Id: cml2html.xel,v 1.8 1999/11/12 20:01:11 $-->
<!--Description: Transform CML to HTML -->
<xsl:stylesheet xmlns:xcl="http://www.w3.org/1999/XSL/Transform"

                xmlns:xt="htCp://www.jclark.com/xt"
                version="1.0"
                extension-element-prefixes="xt">
<xsl:include href="html/cml.xsl"/>
<xsl:include href="html/environment.xsl"/>
<xsl:include href="html/output.xsl"/>
<xel:include href="html/selections.xsl"/>
<xsl:include href="common/identity.xsl"/>
</xsl:stylesheet>
<!--$Id: cml.xel,v 1.13 2000/01/31 Exp $-->
<!--Description: Translate CML element to HTML -->
<!-- Handle case of CML element being the top-level element -->
<xsl:stylesheet
                 xmlUs:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output method="html"/>
  <xsl:template match="/cml">
    <html>
      <head>
        <META http-equiv="Content-Type" content="text/html;
          charset=iso-8859-1"/>
        <title><xsl:value-of select="@title"/></title>
      </head>
      <body>
        <hl>
          <a name="{@name}">
            <xel:value-of select="@title"/>
          </a>
        </hl>
        <xsl:choose>
          <xsl:when test="@action='submit'">
            <form>
              <xsl:attribute name="node_id">
               <xsl:value-of select="@node_id"/>
              </xsl:attribute>
              <xsl:attribute name="action">
                <xsl:value-of select="submit/@target"/>
              </xsl:attribute>
              <xsl:apply-templates/>
              <p>
                <INPUT TYPE="SUBMIT" VALUE="@name"/>
             < /p>
            </form>
          </xsl:when>
```

```
          <xsl:otherwise>
            <div node_id="{@node_id}"
                 name="{@name}">
              <xsl:apply-templates/>
            </div>
          </xsl:otherwise>
      </xsl:choose>
       </body>
     </html>
</xsl:template>

<xsl:template match="cml[@action='submit']">
  <h2> <a name="{@name}">
       <xsl:value-of select="@title"/> </a>
  </h2>
  <form>
    <xsl:attribute name="node_id">
      <xsl:value-of select="@node_id"/>
    </xsl:attribute>
    <xsl:attribute name="action">
      <!-- for rea, we should process submit node to
      cons up target uri -->
      <xsl:value-of select="../submit/@target"/>
    </xsl:attribute>
    <xcl:apply-templates/>
    <p>
      <INPUT TYPE="SUBMIT" VALUE="{@name}"/>
    </P>
  </form>
</xsl:template>
<xsl:template match="cml">
<h2 node_id="{@node_id}">
<a name="{@name}">
   <xsl:value-of select="@title"/> </a>
</h2>
<xsl:apply-templates/>
<xSl:if test="@action='return'">
  <p>
    <a name="{concat('#', /cml/@name)}">
      Back
    </a>
  < /p>
 </xsl:if>
</xsl:template>
<xsl:template match="group">
  <div groupId="{@groupId}"
       modality="{@modality}"
       class="{@class}">
  <xsl:apply-templates/>
  </div>
```

```
</xsl:template>
<xsl:template match="submit"/>
</xsl:stylesheet>
<!--$Id: environment.xsl,v 1.2 2000/02/01 Exp $ -->
<!--Description: Process CML environment constructs -->
<xsl:stylesheet
                xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
   <xsl:template match="final">
     <xsl:apply-templates/>
   </xsl:template>
   <xsl:template match="var">
     <input type="hidden" name="{@name}" value="{@expr}"/>
   </xsl:template>
   <xsl:template match="assign">
     <input name="{@name}" type="hidden">
       <xsl:attribute name="value">
      <xsl:choose>
            <xsl:when test="@expr=''">
             <xsl:value-of select="./node()"/>
           </xsl:when>
           <xsl:otherwise>
             <xsl:value-of select="@expr"/>
           </xsl:otherwise>
         </xsl:choose>
       </xsl:attribute>
     </input>
   </xcl:template>
 <xsl:template match="value">
   <b><xsl:value-of select="@name"/></b>
  </xsl:template>
</xsl:stylesheet>
<!--$Id: output.xsl,v 1.3 1999/11/12 20:07:23 Exp $-->
<!-- Description: Transformation rules for CML gestures that -->
<!-- primarily output information -->
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="message">
    <P>
      <xsl:attribute name="node_id">
        <xsl:value-of select="@node_id"/>
      </xsl:attribute>
      <xsl:apply-templates/>
    </P>
  </xsl:template>
<!-- eventually generate pop-up help via javascript -->
<xsl:template match="help">
    <P>
      <xsl:attribute name="node_id">
        <xsl:value-of select="@node_id"/>
      </xsl:attribute>
      <xsl:apply-templates/>
```

```
    </P>
  </xsl:template>
</xsl:stylesheet>
<!--$Id: selections.xsl,v 1.8 2000/01/31 17:50:34 $-->
<!--Descriptions: Transform CML selection gestures to HTML -->
<xsl:stylesheet
                xmlus:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="menu">
    <xsl:if test="@title!=''">
      <h2>
        <a name="#{@name}">
          <xsl:value-of select="@title"/>
        </a>
      </h2>
    </xsl:if>
    <xsl:apply-templates select="message"/>
    <ol node_id="{@node_id}">
      <xsl:for-each select="choices/choice|choices/default">
        <li>
          <a href="{@value}">
           <xsl:apply-templates/>
          </a>
        </li>
      </xsl:for-each>
    </ol>
</xsl:template>
<xsl:template match="select">
<xsl:apply-templates select="message"/>
 <select name="{@name}">
   <xsl:apply-templates select="choices"/>
 </select>
 <P/>
</xsl:template>
<xsl:template match="choices">
<xsl:apply-templates/>
</xsl:template>
<xsl:template match="choice|default">
<option>
   <xsl:attribute name="value">
     <xsl:value-of select="@value"/>
   </xsl:attribute>
   <xsl:if test="name(.)='default'">
     <xsl:attribute name="checked"/>
   </xsl:if>
   <xsl:apply-templates/>
  </option>
 </xsl:template>
 <xsl:template match="grammar" />
</xsl:stylesheet>
<!--$Id: identity.xsl,v 1.1 1999/11/08 18:05:26 Exp $-->
```

```
<!-- Description: Identity transform for use in other sheets-->
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="*¦@*">
    <xsl:value-of select="."/>
    <xsl:copy>
      <xsl:apply-templates select="@*"/>
      <xsl:apply-templates select="node()"/>
    </xsl:copy>
  </xsl:template>
</xsl:stylesheet>
```

【０１４４】（ｃ）ＨＴＭＬソース

以下は、（ＣＭＬからＨＴＭＬへの）ＸＳＬソースをＣＭＬソース・ページ上に適用することによって得られるＨＴＭＬソース・ページを記述する。得られた歓迎ＧＵＩページをＨＴＭＬブラウザで見たものを、図７から９に示す。

```
<!DOCTYPE html PUBLIC "-/M13C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
<META http-equiv=" Content-Type" content= "text/html; charset=iso-8859-
1 " >
<title>CNN Mobile News</title>
</head>
<body>
<hl>
<a name="cnn">CNN Mobile News</a>
</hl>
<divnode_id="1" name="cnn">
<ol node_id="2">
<li>
<a href="#cnn.query">Select News Stories</a>
</li>
<li>
<a href="#cnn.exit">
     Exit </a>
</li>
<li>
<a href="#cnn.applicationHelp">Help</a>
</li>
</ol>
<h2 node_id="4">
<a name="cnn.applicationHelp">About CNN Mobile</a>
</h2>
   <P node_id=" 5 " >
   This application allows you to select and view CNN news stories
  </P>
<p>
<a name="#cnn">
    Back
        </a>
</p>
  <h2>
<a name="cnn.exit">Exit CNN Mobile News</a>
```

```
</h2>
<form node_id= " 6 " action="http://raman.almaden.ibm.com/cgi-bin/cnn.cg
i?command=exit" >
     <P node id="60">
     Thankyou for using the CNN news service
   </P>
<p>
<INPUT TYPE="SUBMIT" VALUE="cnn.exit">
</p>
</form>
   <div groupId="query" modality="" class="">
     <h2 node_id="8">
<a name="cnn.query">Search CNN Mobile News</a>
</h2>
     <h2>
<a name="#cnn.query.topic">Topic Selection</a>
</h2>
<ol node_id="11">
<li>
<a href="#cnn.query.news"> News </a>
</li>
<li>
<a href="#cnn.query.business"> Business </a>
</li>
<li>
<a href="#cnn.query.sports">
           Sports
        </a>
</li>
<li>
<a href="#cnn.query.travel"> Travel </a>
</li>
<li>
<a href="#cnn.query.weather"> Weather </a>
</li>
<li>
<a href="#cnn.query.show">
           Show Business
        </a>
</li>
</ol>
   <h2>
<a name="cnn.query.news">News Channel</a>
</h2>
<form node_id= " 13 " action="http://raman.almaden.ibm.com/cgi-bin/cnn.cg
i?command=search" >
     <P node_id="9">
         Which part of today's news would you like to read?</P>
<select name="cnn.query.part">
          <option value="h"> Headlines</option>
```

```
          <option value="1"> first story </option>
          <option value="2"> second story </option>
          <option value="3"> third story </option>
        </select>
<p>
</p>
      <P node id="14">
          Which news category would you like to read?
         </P>
<select name= " cnn.query.interest" >
            <option value="business">
              Business
            </option>
            <option value="africa">
              Africa</option>
            <option value="world"> World </option>
            <option value="United states"> United states </option>
            <option value="europe"> Europe </option>
            <option value="Asia"> Asia</option>
            <option value="me"> Middle East</option>
            <option value="america"> America </option>
          </select>
<p>
</p>
    <p>
<INPUT TYPE="SUBMIT" VALUE="cnn.query.news">
</p>
</form>
    <h2>
<a name= " cnn.query.business" >Business Channel</a>
</h2>
<form node_id=" 16 " action="http://raman.almaden.ibm.com/cgi-bin/cnn.cg
i?command=search" >
<P node_id="9">
          Which part of today's news would you like to read?</P>
<select name="cnn.query.part">
          <option value="h"> Headlines</option>
          <option value=" 1 " > first story </option>
          <option value="2"> second story </option>
          <option value="3"> third story </option>
        </select>
<p>
</p>
     <P node_id="17">
         Which business category would you like to read?</P>
<select name="cnn.query.interest" >
    <option value="NEWS"> news </option>
          <option value="IN"> indexes </option>
          <option value="CU"> exchange rates </option>
          <option value="MET"> metals </option>
```

```
        </select>
<p>
</p>
    <p>
<INPUT TYPE="SUBMIT" VALUE=" cnn.query.business" >
</p>
</form>
    <h2>
<a name="cnn.query.weather">Weather Channel</a>
</h2>
<form node_id= " 19 " action="http://raman.almaden.ibm.com/cgi-bin/cnn.c
gi?command=search" >
       <P node id="9">
          Which part of today's news would you like to read?</P>
<select name="cnn.query.part">
          <option value="h"> Headlines</option>
          <option value= " 1 " > first story </option>
          <option value="2"> second story </option>
          <option value="3"> third story </option>
        </select>
<p>
</p>
     <P node_id="20">
         Which region are you interested in?</P>
<select name= "cnn.query.interest" >
          <option value="us"> United states </option>
          <option value="europe">
         Europe
          </option>
          <option value="JP"> Japan </option>
          <option value="AU"> Australia </option>
          <option value="AS"> Asia </option>
        </select>
<p>
</p>
    <p>
<INPUT TYPE="SUBMIT" VALUE="cnn.query.weather">
</p>
</form>
    <h2>
<a name="cnn.query.travel">Travel Section</a>
</h2>
<form node_id=" 22 " action="http://raman.almaden.ibm.com/cgi-bin/cnn.cg
i?command=search" >
     <P node_id="9">
         Which part of today's news would you like to read?</P>
<select name="cnn.query.part">
          <option value="h"> Headlines</option>
          <option value=" 1 " > first story </option>
          <option value="2"> second story </option>
```

```
          <option value="3"> third story </option>
        </select>
<p>
</p>
      <P node_id= " 23 " >
          Which city do you want to visit?</P>
<select name="cnn.query.interest">
          <option value="AMSTERDAM">AMSTERDAM</option>
          <option value= " COPENHAGEN " >COPENHAGEN</option>
          < option value = " HELSINKI " >HELSINKI</option>
          <option value="HONGKONG">HONGKONG</option>
          <option value="LONDON" >LONDON</option>
          <option value= " OSLO " >OSLO</option>
          <option value="PRAGUE">PRAGUE</option>
          <option value= " SINGAPORE " >SINGAPORE</option>
          <option value="STOCKHOLM" >STOCKHOLM</option>
          <option value="SYDNEY">SYDNEY</option>
        </select>
<p>
</p>
    <p>
<INPUT TYPE="SUBMIT" VALUE="cnn.query.travel">
</p>
</form>
    <h2>
<a name="cnn.query.sports">Sports Channel</a>
</h2>
<form node_id= " 25 " action= "http://raman.almaden.ibm.com/cgi-bin/cnn.
cgi?command=search" >
      <P node_id="9">
          Which part of today's news would you like to read?</P>
<select name="cnn.query.part">
          <option value="h"> Headlines</option>
          <option value= " 1 " > first story </option>
          <option value="2"> second story </option>
          <option value="3"> third story </option>
        </select>
<p>
</p>
     <P node_id="26">
     What sports are you interested in?</P>
<select name="cnn.query.interest">
          <option value="AS"> Asia </option>
          <option value="w" > world </option>
          <option value="eu"> europe </option>
          <option value="us"> united states </option>
          <option value="nba"> NBA </option>
          <option value="nhl"> nhl </option>
          <option value="EF"> Europoean football </option>
</select>
```

```
<p>
</p>
    <p>
<INPUT TYPE="SUBMIT" VALUE="cnn.query.sports">
</p>
</form>
  </div>
</div>
</body>
</html>
```

【０１４５】（ｄ）ジェスチャＸＳＬ

以下の例は、ＣＭＬページをジェスチャごとにＷＭＬページにトランスコーディングするのに使用される、ＣＭＬからＷＭＬへのジェスチャ・ベースのＸＳＬ規則を示す。任意の可能なＣＭＬページをトランスコーディングするのに必要なジェスチャ・ベースのトランスコーディング規則がすべてあるわけではない。これは、この方法の例示と見なすべきである。ＸＳＬ構文は、従来のＸＳＬＴ規則に従う。例えば、http://www.w3.org/1999/XSL/Transformを参照されたい。

```
<!-- $Id: cml2html.xsl,v 1.9 2000/02/05 19:32:40 Exp $ -->
<!-- Description: Transform CML to HTML -->
- <xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:xt="http://www.jclark.com/xt" version="1.0" extension-element-pref
ixes="xt">
<xsl:include href="html/cml.xsl" />
<xsl:include href="html/environment.xsl" />
<xsl:include href="html/modality.xsl" />
<xsl:include href="html/output.xsl" />
<xsl:include href="html/selections.xsl" />
<xsl:include href="common/identity.xsl" />
</xsl:stylesheet>
<!--$Id: cml.xsl,v 1.13 2000/01/31 Exp $-->
<!--Description: Translate CML element to HTML -->
<!-- Handle case of CML element being the top-level element -->
<xsl:stylesheet
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="html"/>
<xsl:template match="/cml">
<html>
<head>
<META http-equiv="Content-Type" content="text/html; charset=iso-8859-1"/
>
<title><xsl:value-of select="@title"/></title>
</head>
<body>
<h1>
<a name="{@name}">
<xsl:value-of select="@title"/>
</a>
</h1>
<xsl:choose>
<xsl:when test="@action='submit'">
<form>
<xsl:attribute name="node_id">
```

```
<xsl:value-of select="@node_id"/>
</xsl:attribute>
<xsl:attribute name="action">
<xsl:value-of select="submit/@target"/>
</xsl:attribute>
<xsl:apply-templates/>
<p>
<INPUT TYPE="SUBMIT" VALUE="@name"/>
</p>
</form>
</xsl:when>
<xsl:otherwise>
<div node_id="{@node_id}"
name="{@name}">
<xsl:apply-templates/>
</div>
</xsl:otherwise>
</xsl:choose>
</body>
</html>
</xsl:template>
<xsl:template match="cml[@action='submit']">
<h2> <a name="{@name}">
<xsl:value-of select="@title"/> </a>
</h2>
<form>
<xsl:attribute name="node_id">
<xsl:value-of select="@node_id"/>
</xsl:attribute>
<xsl:attribute name="action">
<!-- for rea, we should process submit node to
cons up target uri -->
<xsl:value-of select="../submit/@target"/>
</xsl:attribute>
<xsl:apply-templates/>
<p>
<INPUT TYPE="SUBMIT" VALUE="{@name}"/>
</p>
</form>
</xsl:template>
<xsl:template match="cml">
<h2 node_id="{@node_id}">
<a name="{@name}">
<xsl:value-of select="@title"/> </a>
</h2>
<xsl:apply-templates/>
<xsl:if test="@action='return'">
<p>
<a name="{concat('#', /cml/@name)}">
Back
```

```
</a>
</p>
</xsl:if>
</xsl:template>
<xsl:template match="group">
<div groupId="{@groupId}"
modality="{@modality}"
class="{@class}">
<xsl:apply-templates/>
</div>
</xsl:template>
<xsl:template match="submit"/>
</xsl:stylesheet>
<!--$Id: environment.xsl,v 1.2 2000/02/01 Exp $ -->
<!--Description: Process CML environment constructs -->
<xsl:stylesheet
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="final">
<xsl:apply-templates/>
</xsl:template>
<xsl:template match="var">
<input type="hidden" name="{@name}" value="{@expr}"/>
</xsl:template>
<xsl:template match="assign">
<input name="{@name}" type="hidden">
<xsl:attribute name="value">
<xsl:choose>
<xsl:when test="@expr=''">
<xsl:value-of select="./node()"/>
</xsl:when>
<xsl:otherwise>
<xsl:value-of select="@expr"/>
</xsl:otherwise>
</xsl:choose>
</xsl:attribute>
</input>
</xsl:template>
<xsl:template match="value">
<b><xsl:value-of select="@name"/></b>
</xsl:template>
</xsl:stylesheet>
<!-- $Id: modality.xsl,v 1.1 2000/02/05 19:32:00 Exp $ -->
<!-- Description: Process CML modality constructs -->
- <xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
- <xsl:template match="modality[@class='visual']">
<xsl:apply-templates />
</xsl:template>
- <xsl:template match="var">
<input type="hidden" name="{@name}" value="{@expr}" />
</xsl:template>
```

```
- <xsl:template match="assign">
- <input name="{@name}" type="hidden">
- <xsl:attribute name="value">
- <xsl:choose>
- <xsl:when test="@expr=''">
<xsl:value-of select="./node()" />
</xsl:when>
- <xsl:otherwise>
<xsl:value-of select="@expr" />
</xsl:otherwise>
</xsl:choose>
</xsl:attribute>
</input>
</xsl:template>
- <xsl:template match="value">
- <b>
<xsl:value-of select="@name" />
</b>
</xsl:template>
</xsl:stylesheet>
<!--$Id: output.xsl,v 1.3 1999/11/12 20:07:23 Exp $-->
<!-- Description: Transformation rules for CML gestures that -->
<!-- primarily output information -->
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="message">
<P>
<xsl:attribute name="node_id">
<xsl:value-of select="@node_id"/>
</xsl:attribute>
<xsl:apply-templates/>
</P>
</xsl:template>
<!-- eventually generate pop-up help via javascript -->
<xsl:template match="help">
<P>
<xsl:attribute name="node_id">
<xsl:value-of select="@node_id"/>
</xsl:attribute>
<xsl:apply-templates/>
</P>
</xsl:template>
</xsl:stylesheet>
<!--$Id: selections.xsl,v 1.8 2000/01/31 17:50:34 $-->
<!--Descriptions: Transform CML selection gestures to HTML -->
<xsl:stylesheet
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="menu">
<xsl:if test="@title!=''">
<h2>
<a name="#{@name}">
```

```
<xsl:value-of select="@title"/>
</a>
</h2>
</xsl:if>
<xsl:apply-templates select="message"/>
<ol node_id="{@node_id}">
<xsl:for-each select="choices/choice|choices/default">
<li>
<a href="{@value}">
<xsl:apply-templates/>
</a>
</li>
</xsl:for-each>
</ol>
</xsl:template>
<xsl:template match="select">
<xsl:apply-templates select="message"/>
<select name="{@name}">
<xsl:apply-templates select="choices"/>
</select>
<p/>
</xsl:template>
<xsl:template match="choices">
<xsl:apply-templates/>
</xsl:template>
<xsl:template match="choice|default">
<option>
<xsl:attribute name="value">
<xsl:value-of select="@value"/>
</xsl:attribute>
<xsl:if test="name(.)='default'">
<xsl:attribute name="checked"/>
</xsl:if>
<xsl:apply-templates/>
</option>
</xsl:template>
<xsl:template match="grammar" />
</xsl:stylesheet>

<!--$Id: identity.xsl,v 1.1 1999/11/08 18:05:26 Exp $-->
<!-- Description: Identity transform for use in other sheets-->
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="*|@*">
<xsl:value-of select="."/>
<xsl:copy>
<xsl:apply-templates select="@*"/>
<xsl:apply-templates select="node()"/>
</xsl:copy>
</xsl:template>
</xsl:stylesheet>
```

【０１４６】（ｅ）ＷＭＬソース

以下は、（ＣＭＬからＷＭＬへの）ＸＳＬソースをＣＭＬソース・ページ上に適用することによって得られるＷＭＬソース・ページを記述する。得られた歓迎ＧＵＩページをＷＭＬブラウザで見たものを、図１０に示す。

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN" "http://www.wapforum
.org/DTD/wml_1.1.
<wml>
<template>
<do type="prev" label="Back">
<prev/>
</do>
</template>
<card id="cnn.command" title="cnn.command">
<p>
<select name="cnn.command">
<option onpick="#cnn.query">Select News Stories</option>
<option onpick="#cnn.exit">
Exit </option>
<option onpick="#cnn.applicationHelp">Help</option>
</select>
</p>
</card>
<card id="cnn.applicationHelp" title="cnn.applicationHelp">
<p>
This application allows you to select and view CNN news stories
</p>
</card>
<card id="cnn.exit" title="cnn.exit">
<p>
Thankyou for using the CNN news service
</p>
<p align="center">
<a href="cnn.wmls#submit()"/>
</p>
</card>
<card id="cnn.query" title="cnn.query">
<p>
<select name="cnn.query">
<option onpick="#cnn.query.news"> News </option>
<option onpick="#cnn.query.business"> Business </option>
<option onpick="#cnn.query.sports">
Sports
</option>
<option onpick="#cnn.query.travel"> Travel </option>
<option onpick="#cnn.query.weather"> Weather </option>
<option onpick="#cnn.query.show">
Show Business
</option>
</select>
</p>
```

```
</card>
<card id="cnn.query.news" title="cnn.query.news">
<p>
Which part of today's news would you like to read?<select name="cnn.quer
y.part">
<option value="h" onpick="cnn.wmls#submit()"> Headlines</option>
<option value="1" onpick="cnn.wmls#submit()"> first story </option>
<option value="2" onpick="cnn.wmls#submit()"> second story </option>
<option value="3" onpick="cnn.wmls#submit()"> third story </option>
</select>
</p>
<p>
Which news category would you like to read?
<select name="cnn.query.interest">
<option value="business" onpick="cnn.wmls#submit()">
Business
</option>
<option value="africa" onpick="cnn.wmls#submit()">
Africa</option>
<option value="world" onpick="cnn.wmls#submit()"> World </option>
<option value="United states" onpick="cnn.wmls#submit()"> United states
</option>
<option value="europe" onpick="cnn.wmls#submit()"> Europe </option>
<option value="Asia" onpick="cnn.wmls#submit()"> Asia</option>
<option value="me" onpick="cnn.wmls#submit()"> Middle East</option>
<option value="america" onpick="cnn.wmls#submit()"> America </option>
</select>
</p>
<p align="center">
<a href="cnn.wmls#submit()"/>
</p>
</card>
<card id="cnn.query.business" title="cnn.query.business">
<p>
Which part of today's news would you like to read?<select name="cnn.quer
y.part">
<option value="h" onpick="cnn.wmls#submit()"> Headlines</option>
<option value="1" onpick="cnn.wmls#submit()"> first story </option>
<option value="2" onpick="cnn.wmls#submit()"> second story </option>
<option value="3" onpick="cnn.wmls#submit()"> third story </option>
</select>
</p>
<p>
Which business category would you like to read?<select name="cnn.query.i
nterest"
<option value="NEWS" onpick="cnn.wmls#submit()"> news </option>
<option value="IN" onpick="cnn.wmls#submit()"> indexes </option>
<option value="CU" onpick="cnn.wmls#submit()"> exchange rates </option>
<option value="MET" onpick="cnn.wmls#submit()"> metals </option>
</select>
```

```
</p>
<p align="center">
<a href="cnn.wmls#submit()"/>
</p>
</card>
<card id="cnn.query.weather" title="cnn.query.weather">
<p>
Which part of today's news would you like to read?<select name="cnn.quer
y.part">
<option value="h" onpick="cnn.wmls#submit()"> Headlines</option>
<option value="1" onpick="cnn.wmls#submit()"> first story </option>
<option value="2" onpick="cnn.wmls#submit()"> second story </option>
<option value="3" onpick="cnn.wmls#submit()"> third story </option>
</select>
</p>
<p>
Which region are you interested in?<select name="cnn.query.interest">
<option value="us" onpick="cnn.wmls#submit()"> United states </option>
<option value="europe" onpick="cnn.wmls#submit()">
Europe
</option>
<option value="JP" onpick="cnn.wmls#submit()"> Japan </option>
<option value="AU" onpick="cnn.wmls#submit()"> Australia </option>
<option value="AS" onpick="cnn.wmls#submit()"> Asia </option>
</select>
</p>
<p align="center">
<a href="cnn.wmls#submit()"/>
</p>
</card>
<card id="cnn.query.travel" title="cnn.query.travel">
<p>
Which part of today's news would you like to read?<select name="cnn.quer
y.part">
<option value="h" onpick="cnn.wmls#submit()"> Headlines</option>
<option value="1" onpick="cnn.wmls#submit()"> first story </option>
<option value="2" onpick="cnn.wmls#submit()"> second story </option>
<option value="3" onpick="cnn.wmls#submit()"> third story </option>
</select>
</p>
<p>
Which city do you want to visit?<select name="cnn.query.interest">
<option value="AMSTERDAM" onpick="cnn.wmls#submit()">AMSTERDAM</option>
<option value="COPENHAGEN" onpick="cnn.wmls#submit()">COPENHAGEN</option
>
<option value="HELSINKI" onpick="cnn.wmls#submit()">HELSINKI</option>
<option value="HONGKONG" onpick="cnn.wmls#submit()">HONGKONG</option>
<option value="LONDON" onpick="cnn.wmls#submit()">LONDON</option>
<option value="OSLO" onpick="cnn.wmls#submit()">OSLO</option>
<option value="PRAGUE" onpick="cnn.wmls#submit()">PRAGUE</option>
```

```
<option value="SINGAPORE" onpick="cnn.wmls#submit()">SINGAPORE</option>
<option value="STOCKHOLM" onpick="cnn.wmls#submit()">STOCKHOLM</option>
<option value="SYDNEY" onpick="cnn.wmls#submit()">SYDNEY</option>
</select>
</p>
<p align="center">
<a href="cnn.wmls#submit()"/>
</p>
</card>
<card id="cnn.query.sports" title="cnn.query.sports">
<p>
Which part of today's news would you like to read?<select name="cnn.quer
y.part">
<option value="h" onpick="cnn.wmls#submit()"> Headlines</option>
<option value="1" onpick="cnn.wmls#submit()"> first story </option>
<option value="2" onpick="cnn.wmls#submit()"> second story </option>
<option value="3" onpick="cnn.wmls#submit()"> third story </option>
</select>
</p>
<p>
What sports are you interested in?<select name="cnn.query.interest">
<option value="AS" onpick="cnn.wmls#submit()"> Asia </option>
<option value="w" onpick="cnn.wmls#submit()"> world </option>
<option value="eu" onpick="cnn.wmls#submit()"> europe </option>
<option value="us" onpick="cnn.wmls#submit()"> united states </option>
<option value="nba" onpick="cnn.wmls#submit()"> NBA </option>
<option value="nhl" onpick="cnn.wmls#submit()"> nhl </option>
<option value="EF" onpick="cnn.wmls#submit()"> Europoean football </opti
on>
</select>
</p>
<p align="center">
<a href="cnn.wmls#submit()"/>
</p>
</card>
</wml>
```

【０１４７】（ｆ）ジェスチャＸＳＬ

以下の例は、ＣＭＬページをジェスチャごとにVoiceＸＭＬページにトランスコーディングするのに使用される、ＣＭＬからVoiceＸＭＬへのジェスチャ・ベースのＸＳＬ規則を例示する。任意の可能なＣＭＬページをトランスコーディングするのに必要なジェスチャ・ベースのトランスコーディング規則がすべてあるわけではない。これは、この方法の例示と見なすべきである。ＸＳＬ構文は、従来のＸＳＬＴ規則に従う。例えば、http://www.w3.org/1999/XSL/Transformを参照されたい。

```
<!-- cml2wml.xsl -->
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Tran
sform">
<!--
<xsl:output method="html" indent="yes"/>
-->
<xsl:output method="xml" indent="yes" media-type="text/xml"/>
<xsl:template match="/cml">
<xsl:text disable-output-escaping="yes">
```

```
&lt;!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN" "http://www.wapfo
rum.org/DTD/wml_1
</xsl:text>
<wml>
<template>
<do type="prev" label="Back">
<prev/>
</do>
</template>
<xsl:apply-templates/>
</wml>
</xsl:template>
<xsl:template match="cml">
<xsl:choose>
<xsl:when test="menu">
<!-- to avoid <card><card>..</card></card> -->
<card>
<xsl:attribute name="id">
<xsl:value-of select="@name"/>
</xsl:attribute>
<xsl:attribute name="title">
<xsl:value-of select="@name"/>
</xsl:attribute>
<p><select>
<xsl:attribute name="name">
<xsl:value-of select="menu/@name"/>
</xsl:attribute>
<xsl:apply-templates select="menu/message"/>
<xsl:for-each select="menu/choices/choice | menu/choices/default">
<option>
<xsl:attribute name="value">
<xsl:value-of select="@value"/>
</xsl:attribute>
<xsl:attribute name="onpick">#<xsl:value-of select="@value"/></xsl:attri
bute>
<xsl:call-template name="lex"/></option>
</xsl:for-each>
</select>
</p>
</card>
</xsl:when>
<xsl:otherwise>
<card>
<xsl:attribute name="id">
<xsl:value-of select="@name"/>
</xsl:attribute>
<xsl:attribute name="title">
<xsl:value-of select="@name"/>
</xsl:attribute>
<xsl:apply-templates/>
```

```
</card>
</xsl:otherwise>
</xsl:choose>
</xsl:template>
<xsl:template match='cml[@action="submit"]'>
<card>
<xsl:attribute name="id">
<xsl:value-of select="@name"/>
</xsl:attribute>
<xsl:attribute name="title">
<xsl:value-of select="@name"/>
</xsl:attribute>
<xsl:apply-templates/>
<p align="center">
<a>
<xsl:attribute name="href">
<xsl:value-of select="/cml/@name"/>.wmls#submit()</xsl:attribute>
</a>
</p>
</card>
</xsl:template>
<xsl:template match='select'>
<p>
<xsl:apply-templates select="message"/>
<select>
<xsl:attribute name="name">
<xsl:value-of select="@name"/>
</xsl:attribute>
<xsl:for-each select="choices/choice ¦ choices/default">
<option>
<xsl:attribute name="value">
<xsl:value-of select="@value"/>
</xsl:attribute>
<xsl:attribute name="onpick">
<xsl:value-of select="/cml/@name"/>.wmls#submit()</xsl:attribute>
<xsl:call-template name="lex"/></option>
</xsl:for-each>
</select>
</p>
</xsl:template>
<xsl:template match="menu">
<card>
<xsl:attribute name="id">
<xsl:value-of select="@name"/>
</xsl:attribute>
<xsl:attribute name="title">
<xsl:value-of select="@name"/>
</xsl:attribute>
<p>
<select>
```

```
<xsl:attribute name="name">
<xsl:value-of select="@name"/>
</xsl:attribute>
<xsl:apply-templates select="message"/>
<xsl:for-each select="choices/choice ¦ choices/default">
<option>
<xsl:attribute name="value">
<xsl:value-of select="@value"/>
</xsl:attribute>
<xsl:attribute name="onpick">#<xsl:value-of select="@value"/></xsl:attri
bute>
<xsl:call-template name="lex"/></option>
</xsl:for-each>
</select>
</p>
</card>
</xsl:template>
<xsl:template name="lex">
<xsl:for-each select="node()">
<xsl:if test="position()=last()">
<xsl:value-of select="current()"/>
</xsl:if>
</xsl:for-each>
</xsl:template>
<!-- explicitly remove segment -->
<xsl:template match="submit"/>
<xsl:template match="message"/>
</xsl:stylesheet>
```

【０１４８】（ｇ）VoiceＸＭＬを作成するためのＸＳＬソース　以下に、VoiceＸＭＬソース・ページを作成するのに使用されるＸＳＬソース・コードを記述する。

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Tran
sform">
<xsl:output method="html"/>
<xsl:template match="/cml">
<vxml>
<xsl:apply-templates/>
</vxml>
</xsl:template>
<xsl:template match="menu">
<menu>
<xsl:apply-templates select="message"/>
<xsl:attribute name="id">
<xsl:value-of select="@name"/>
</xsl:attribute>
<xsl:attribute name="node_id">
<xsl:value-of select="@node_id"/>
</xsl:attribute>
<xsl:apply-templates select="message"/>
<prompt> Say one of <enumerate/> </prompt>
<xsl:for-each select="choices/choice¦choices/default">
```

```
<choice>
<xsl:attribute name="next">#<xsl:value-of select="@value"/></xsl:attribu
te>
<xsl:apply-templates/>
</choice>
</xsl:for-each>
</menu>
</xsl:template>
<xsl:template match="cml[@action='return']">
<form>
<xsl:attribute name="id">
<xsl:value-of select="@name"/>
</xsl:attribute>
<xsl:attribute name="node_id">
<xsl:value-of select="@node_id"/>
</xsl:attribute>
<xsl:apply-templates/>
<block><goto>
<xsl:attribute name="next">#<xsl:value-of select="/cml/menu/@name"/></xs
l:attribut
</goto></block>
</form>
</xsl:template>
<xsl:template match="cml[@action='submit']">
<form>
<xsl:attribute name="id">
<xsl:value-of select="@name"/>
</xsl:attribute>
<xsl:attribute name="node_id">
<xsl:value-of select="@node_id"/>
</xsl:attribute>
<xsl:apply-templates/>
<block>
<goto next="http://raman.almaden.ibm.com/cgi-bin/cnn.cgi">
<xsl:if test="select[@name]">
<xsl:for-each select="select">
<xsl:attribute name="submit">
<xsl:value-of select="@name"/>
</xsl:attribute>
</xsl:for-each>
</xsl:if>
</goto>
</block>
</form>
</xsl:template>
<xsl:template match="select">
<field>
<xsl:attribute name="name">
<xsl:value-of select="@name"/>
</xsl:attribute>
```

```
<xsl:attribute name="node_id">
<xsl:value-of select="../@node_id"/>
</xsl:attribute>
<xsl:if test="message">
<prompt>
<xsl:value-of select="message"/>
Say one of <enumerate/>
</prompt>
</xsl:if>
<grammar>
<xsl:for-each select="choices/choice|choices/default">
<xsl:call-template name="lex"/>
<xsl:if test="following-sibling::choice">|</xsl:if>
</xsl:for-each>
</grammar>
</field>
</xsl:template>
<xsl:template match="message">
<field><prompt>
<xsl:attribute name="node_id">
<xsl:value-of select="@node_id"/>
</xsl:attribute>
<xsl:apply-templates/>
</prompt>
</field>
</xsl:template>
<xsl:template match="help">
<help>
<xsl:attribute name="node_id">
<xsl:value-of select="@node_id"/>
</xsl:attribute>
<xsl:apply-templates/>
</help>
</xsl:template>
<xsl:template match="grammar"/>
<xsl:template match="submit"/>
<xsl:template name="lex">
<xsl:for-each select="node()">
<xsl:if test="position()=last()">
<xsl:value-of select="current()"/>
</xsl:if>
</xsl:for-each>
</xsl:template>
</xsl:stylesheet>
```

【０１４９】（ｈ）VoiceＸＭＬソース

以下に、（ＣＭＬからVoiceＸＭＬへの）ＸＳＬソースをＣＭＬソース・ページ上に適用することによって得られるVoiceＸＭＬソース・ページを記述する。得られる歓迎スピーチ・ダイアログは、VoiceＸＭＬブラウザで呈示されるとき、異なるオプション間で音声によって選択するためのダイアログを最初にユーザに呈示する。

```
<vxml>
<menu id="cnn_command" node_id="2">
```

```
<prompt> Say one of <enumerate></enumerate></prompt><choice next="#cnn_q
uery">Select News
Exit </choice><choice next="#cnn_applicationHelp">Help</choice>
</menu>
<form id="cnn_applicationHelp" node_id="4">
<field><prompt node_id="5">
This application allows you to select and view CNN news stories
</prompt></field>
<block><goto next="#cnn"></goto></block>
</form>
<form id="cnn_exit" node_id="6">
<field><prompt node_id="60">
Thankyou for using the CNN news service
</prompt></field>
<block><goto next="http://raman.almaden.ibm.com/cgi-bin/cnn.cgi"></goto>
</block>
</form>
<menu id="cnn_query" node_id="11">
<prompt> Say one of <enumerate></enumerate></prompt><choice next="#cnn_q
uery_news"> News <
Sports
</choice><choice next="#cnn_query_travel"> Travel </choice><choice next=
"#cnn_qu
Show Business
</choice>
</menu>
<form id="cnn_query_news" node_id="13">
<field name="cnn_query_part" node_id="13"><prompt>
Which part of today's news would you like to read?</prompt><grammar> Hea
dlines|
<field name="cnn_query_interest" node_id="13"><prompt>
Which news category would you like to read?
</prompt><grammar>
Business
|
Africa| World | United states | Europe | Asia| Middle East| America </gr
amma
<block><goto next="http://raman.almaden.ibm.com/cgi-bin/cnn.cgi" submit=
"cnn_query_int
</form>
<form id="cnn_query_business" node_id="16">
<field name="cnn_query_part" node_id="16"><prompt>
Which part of today's news would you like to read?</prompt><grammar> Hea
dlines|
<field name="cnn_query_interest" node_id="16"><prompt>
Which business category would you like to read?</prompt><grammar> news |
 indexes
<block><goto next="http://raman.almaden.ibm.com/cgi-bin/cnn.cgi" submit=
"cnn_query_int
</form>
```

```
<form id="cnn_query_weather" node_id="19">
<field name="cnn_query_part" node_id="19"><prompt>
Which part of today's news would you like to read?</prompt><grammar> Hea
dlines¦
<field name="cnn_query_interest" node_id="19"><prompt>
Which region are you interested in?</prompt><grammar> United states ¦
Europe
¦ Japan ¦ Australia ¦ Asia </grammar></field>
<block><goto next="http://raman.almaden.ibm.com/cgi-bin/cnn.cgi" submit=
"cnn_query_int
</form>
<form id="cnn_query_travel" node_id="22">
<field name="cnn_query_part" node_id="22"><prompt>
Which part of today's news would you like to read?</prompt><grammar> Hea
dlines¦
<field name="cnn_query_interest" node_id="22"><prompt>
Which city do you want to visit?</prompt><grammar>AMSTERDAM¦COPENHAGEN¦H
ELSINKI¦
<block><goto next="http://raman.almaden.ibm.com/cgi-bin/cnn.cgi" submit=
"cnn_query_int
</form>
<form id="cnn_query_sports" node_id="25">
<field name="cnn_query_part" node_id="25"><prompt>
Which part of today's news would you like to read?</prompt><grammar> Hea
dlines¦
<field name="cnn_query_interest" node_id="25"><prompt>
What sports are you interested in?</prompt><grammar> Asia ¦ world ¦ euro
pe ¦ uni
<block><goto next="http://raman.almaden.ibm.com/cgi-bin/cnn.cgi" submit=
"cnn_query_int
</form>
</vxml>
```

【０１５０】（iii）密なマルチモーダル・ブラウジングおよびマルチデバイス・ブラウジング

上に述べ、かつ、以下により詳細に述べるように、異なるモーダル性を密に同期させることができる。例えば、同じ装置上の音声およびＧＵＩとすることもでき、ＨＴＭＬまたはWMLブラウザ上のＧＵＩと同期をとられた電話上の音声とすることもできるなどである。

【０１５１】Ｍ．表面変更

モーダル性特定の表面的コンテンツまたはパラメータを、モーダル性特定のＸＭＬ構文を使用して追加することができる。モーダル性特定のジェスチャは、モーダル性修飾詞を有するモーダル性特定のＸＭＬ構文を使用して追加することができる。その他のモーダル性は、これらのコンポーネントを無視するか、または他のもの（例えばキャプション）で置換することができる。

【０１５２】（ｉ）モーダル性特定の情報

ＣＭＬは、対話論理および会話型アプリケーション・フローを指定するための宣言的かつモーダル性独立のマークアップ言語に設計される。しかし、アプリケーション作成者がそのうちに、カスタム・プレゼンテーションを達成するためにＣＭＬアプリケーションにモーダル性特定のコンテンツを追加したいと思うであろうことは理解される。ＣＭＬは、特定のモーダル性用とされるマークアップの断片をカプセル化するのに使用される要素modalityによって、これを可能にする。このようなモーダル性特定の断片は、指定されたモーダル性にしか表れないことになることに留意されたい。したがって、作成者は、このようなモーダル性特定の断片を、絶対に必要と思われる場合、さらには作成者が他のモーダル性用の代替断片を提供するかまたは他のどんなモーダル性も気にかけない場合だけに使用することが推奨される。ＸＭＬ属性のclassおよびmoduleによって限定される要素modalityは、以下のように定義される。

Class　この断片が適用されるモーダル性のクラスを指定する。

module　この断片を受け入れることのできるマークアッ

プ言語モジュールを指定する。　　なるＨＴＭＬ特定の断片である。

【０１５３】以下は、視覚表現までに通過されることに

```
<modality class="visual" module="html-basic">
<LINK REL="stylesheet"
HREF="cnn.css"
TYPE="text/css"/>
</modality>
      The following is an example of a cosmetized CML page:
<!--$Id: cnn.cml,v 1.21 2000/02/05 20:08:27 Exp $-->
<!--Description: CNN Mobile In cml -->
<cml name="cnn"
   node_id="1"
   title="CNN Mobile News">
  <modality class="visual" module="html-basic">
    <LINK REL="stylesheet"
          HREF="cnn.css"
          TYPE="text/css"/>
  </modality>
  <modality class="visual" module="html">
  <TABLE BORDER="0" WIDTH="600" CELLSPACING="0" CELLPADDING="0"><TR>
      <TD WIDTH="122"VALIGN="TOP"><a H
        <IMG SRC="http://cnn.com/images/1999/10/cnnstore.gif"
     WIDTH="120" HEIGHT="60" BORDER="1" AL
   <TD WIDTH="8" VALIGN="TOP"><a HREF="http://cnn.com/ads/
               e.market/">
            <IMG SRC="http://cnn.com/images/1998/05/homepage/ad.
               info.gif" WIDTH="7" HEIGHT="62" BORDER=
      <TD WIDTH="470" VALIGN="TOP">
         <a
             HREF= " http: / /cnn. com/event.ng/Type=click%26RunID=
               11875%26Prof ileID=34%2 6AdID=13042%2 6Group:
    target="_top">
                 <img src="http://cnn.com/ads/advertiser/promo/
                intercompany_onair/9907/onair_egg_cnn.giff"
                     border="0'' height="60" width="468" alt="Get to the
                  point news!"/>
         </a>
<table width="100%" cellpadding="0" cellspacing="0"
                                  border="0"><tr><td align="right"''><font
 face="verdana,ARIAL,sans-serif" size="1"><a
     </TD></TR></TABLE>
  </modality>
  <modality class="speech" module="vxml''>
       <block>
         Shop CNN for all your information needs!
         </block>
  </modality>
  <menu name="cnn.command"
             node_id="2">
      <choices node_id="3" >
```

```
        <default value=卅#cnn.query''>Select News Stories</default>
        <choice value="#cnn.exit"
                require_confirmation="true">
          Exit </choice>
  <choice value="#cnn.applicationHelp">Help</choice>
    </choices>
   </menu>
   <cml name="cnn.applicationHelp"
        title="About CNN Mobile"
        node_id=''4"
        action="return">
  <message
                    node id="5" >
       This application allows you to select and view CNN news stories
  </message>
</cml>
<cml name="cnn.exit"
         node_id="6"
         title="Exit CNN Mobile News"
         action="submit">
  <message node_id="60">
       Thankyou for using the CNN news service
  </message>
</cml>
<group node_id="7"
        groupId="query">
   <cml name="cnn.query"
        title="Search CNN Mobile News"
        node_id="58">
       <menu name="cnn.query.topic"
              node_id="11"
              title="Topic Selection">
        <choices node_id="12'' >
           <choice value="#cnn.query.news"> News </choice>
           <choice value="#cnn.query.business"> Business </choice>
           <choice value="#cnn.query.sports">
                <grammar> (sport ¦ sports" </grammar>
                Sports
                </choice>
                <choice value="#cnn.query.travel"> Travel </choice>
                <choice value="#cnn.query.weather"> Weather </choice>
                <choice value=*#cnn.query.show">
                        <grammar > show [business] </grammar>
                        Show business
                </choice>
       </choices>
     </menu>
</cml>
<cml name=*cnn.query.news*
     title="News Channel"
```

```
     node_id=*13"
     action="submit">
<select name=*cnn.query.part">
    <message node_id=*9* >
     Which part of today's news would you like to read?</message>
    <choices
        node_id="10" >
      <choice value="h"> Headlines</choice>
      <choice value="1"> first story </choice>
      <choice value="2"> second story </choice>
      <choice value="3"> third story </choice>
     </choices>
   </select>
<select name="cnn.query.interest">
  <message node_id="14">
    Which news category would you like to read?
  </message>
  <choices node_id="15" >
    <choice value="business">
      <grammar type="text/jsgf">
        business {BIZ}</grammar>
     Business
    </choice>
    <choice value="africa">
      Africa</choice>
    <choice value="world"> World </choice>
    <choice value="United states"> United states </choice>
            <choice value="europe"> Europe </choice>
    <choice value="Asia"> Asia</choice>
    <choice value="me"> Middle East</choice>
    <choice value="america"> America </choice>
  </choices>
</select>
</cml>
<cml name="cnn.query business"
     title="Business Channel"
     action="submit"
     node_id="16" >
  <select name="cnn.query.part">
    <message node_id="9" >
      Which part of today*s news would you like to read?</message>
  <choices
          node_id="10" >
   <choice value="h"> Headlines</choice>
   <choice value="1"> first story </choice>
   <choice value="2"> second story </choice>
   <choice value="3"> third story </choice>
  </choices>
 </select>
 <select name="cnn.query.interest">
```

```
   <message node_id="17">
     Which business category would you like to read?</message>
  <choices node_id="18">
    <choice value="NEWS"> news </choice>
    <choice value="IN"> indexes </choice>
    <choice value="CU"> exchange rates </choice>
    <choice value="MET"> metals </choice>
  </choices>
</select>
</cml>
<cml name="cnn.query.weather"
     title="Weather Channel"
     action="submit"
     node_id="19" >
 <select name="cnn.query.part">
   <message node_id="9" >
     Which part of today*s news would you like to read?</message>
  <choices node_id="10" >
    <choice value="h"> Headlines</choice>
    <choice value="1"> first story </choice>
   <choice value="2"> second story </choice>
    <choice value="3"> third story </choice>
  </choices>
 </select>
 <select name="cnn.query.interest">
 <message node_id="20">
     Which region are you interested in?</message>
  <choices node_id="21">
    <choice value="us"> United states </choice>
    <choice value="europe">
      <grammar type="text/jsgf"> (euro | Europe) </grammar>
      Europe
    </choice>
    <choice value="JP"> Japan </choice>
    <choice value="AU"> Australia </choice>
    <choice value="AS"> Asia </choice>
 </choices>
 </select>
</cml>
<cml name="cnn.query.travel"
     title="Travel Section"
    action="submit"
    node_id="22" >
 <select name="cnn.query.part">
   <message node_id="9" >
     Which part of today*s news would you like to read?</message>
  <choices
           node_id="10">
    <choice value="h"> Headlines</choice>
    <choice value="1"> first story </choice>
```

```
    <choice value="2"> second story </choice>
    <choice value="3"> third story </choice>
  </choices>
</select>
 <select name="cnn.query.interest">
   <message node_id="23">
     Which city do you want to visit?</message>
  <choices node_id="24">
    <choice value="AMSTERDAM">AMSTERDAM</choice>
    <choice value="COPENHAGEN">COPENHAGEN</choice>
    <choice value="HELSINKI">HELSINKI</choice>
    <choice value="HONGKONG">HONGKONG</choice>
    <choice value="LONDON">LONDON</choice>
    <choice value="OSLO">OSLO</choice>
    <choice value="PRAGUE">PRAGUE</choice>
    <choice value="SINGAPORE">SINGAPORE</choice>
    <choice value="STOCKHOLM">STOCKHOLM</choice>
    <choice value="SYDNEY">SYDNEY</choice>
  </choices>
 </select>
</cml>
<cml name="cnn.query.sports"
     action="submit"
     title="Sports Channel"
     node_id="25">
 <select name="cnn.query.part">
 <message node_id="9">
     Which part of today's news would you like to read?</message>
  <choices
           node_id="10" >
    <choice value="h"> Headlines</choice>
    <choice value="1"> first story </choice>
    <choice value="2"> second story </choice>
    <choice value="3"> third story </choice>
  </choices>
</select>
 <select name="cnn.query.interest">
   <message node_id="26">
     What sports are you interested in?</message>
  <choices node_id="27">
    <choice value="AS"> Asia </choice>
    <choice value="w"> world </choice>
    <choice value="eu"> europe </choice>
    <choice value="us"> united states </choice>
    <choice value="nba"> NBA </choice>
    <choice value="nhl"> nhl </choice>
    <choice value="EF"> Europoean football </choice>
  </choices>
 </select>
</cml>
```

```
 <submit target="http://raman.almaden.ibm.com/cgi-bin/cnn.cgi">
  <message node_id="28">
    executing <value name="cnn.command"/>
    for <value name="cnn.query.part"/>
    stories about <value name="cnn.query.interest"/>
    from topic <value name="cnn.query.topic"/>
  </message>
  <env name="cnn.command"/>
  <env name="cnn.query.topic"/>
  <env name="cnn.query.interest"/>
  <env name="cnn.query.part"/>
 </submit>
</group>
 <submit target=ﾈ'http://raman.almaden.ibm.com/cgi-bin/cnn.cgi">
 </submit>
</cml>
```

【０１５４】以下に、（ＣＭＬからＨＴＭＬへの）ＸＳＬソースをＨＴＭＬによって表面変更されたＣＭＬソース・ページ上に適用することによって得られるＨＴＭＬソース・ページを記述する。得られた歓迎ＧＵＩページをＨＴＭＬブラウザで見たものを、図１１に示す。表面変更は、表面変更されないページと比較するときにはっきりと見ることができる。これは、意のままにページを表面変更する可能性を示す。この場合もやはり、すべてのケースを考慮しているわけではないが、これは、この手法を明瞭に示している。

【０１５５】以下は、得られた表面変更済みのＨＴＭＬソース・ページに関連するコードである。

```
<!DOCTYPE html PUBLIC"-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
<META http-equiv="Content-Type" content="text/html; charset=iso
-8859-1">
<title>CNN Mobile News</title>
</head>
<body>
<a name="cnn">CNN Mobile News</a>
</hl>
<div node_id="1" name="cnn">
     <LINK REL="stylesheet" HREF="cnn.css" TYPE="text/css">
 <TABLE BORDER="0" WIDTH="600" CELLSPACING="0" CELLPADDING="0">
<TR>
<TD WIDTH="122"VALIGN="TOP"><a HREF="http://cgi.cnn.com/cgi-bin/redir ec
t?cnn_store">
<IMG SRC="http://cnn.com/images/1999/10/cnnstore.gif" WIDT H="120" HEIGH
T="60" BORDER="1" ALT="CNN Store"></a></TD>
<TD WIDTH="8" VALIGN="TOP"><a HREF="http://cnn.com/ads/
e.market/">
    <IMG SRC="http://cnn.com/images/1998/05/homepage/ad.info.g if" WIDTH
="7" HEIGHT="62" BORDER="0" ALT="ad info"></a></TD>
    <TD WIDTH="470" VALIGN="TOP">
      <a HREF="http://cnn.com/event.ng/Type=click%26RunID=11875%26 Profi
leID=34%26AdID=13042%26GroupID=15%26FamilyID=1099%26TagValues=4.8.249.43
5.594.606%26Redirect=http:%2F%2Fwww.cnn.com%2FHLN%2Findex_pgm.htm" targe
t="_top">
            <img src="http://cnn.com/ads/advertiser/promo/inter company_
```

```
onair/9907/onair_egg_cnn.gif" border="0" height="60" width="468" alt=
"Get to the point news!">
          </a>
<table width="100%" cellpadding="0" cellspacing="0" border="0">
<tr>
<td align="right"><font face="verdana,ARIAL,sans-serif" size="1"><a href
="http://cnn.com/event.ng/Type=click%26RunID=11875%26ProfileID=
34%26AdID=13042%26GroupID=15%26FamilyID=1099%26TagValues=4.8.249.435.
594.60 6%26Redirect=http:%2F%2Fwww.cnn.com%2FHLN%2Findex_pgm.htm target=
_top"
>Get to the point news!</a></font></td>
</tr>
</table>
</TD>
</TR>
</TABLE>
 <ol node_id="2">
<li>
<a href="#cnn.query">Select News Stories</a>
</li>
<li>
<a href="#cnn.exit">
        Exit </a>
</li>
<li>
<a href="#cnn.applicationHelp">Help</a>
</li>
</ol>
  <h2 node_id="4">
<a name="cnn.applicationHelp">About CNN Mobile</a>
</h2>
    <P node_id="5">
      This application allows you to select and view CNN news stories
    </P>
  <p>
<a href="#cnn">
          Back
        </a>
</p>
  <h2>
<a name="cnn.exit">Exit CNN Mobile News</a>
</h2>
<form node_id="6" action="http://raman.almaden.ibm.com/cgi-bin/
cnn.cgi">
    <P node_id="60">
      Thankyou for using the CNN news service
    </P>
  <p>
<INPUT TYPE="SUBMIT" VALUE="cnn.exit">
< /p>
```

```
</form>
  <div groupId="query" modality="" class="">
    <h2 node_id="8">
<a name="cnn.query">Search CNN Mobile News</a>
</h2>
      <h2>
<a name="#cnn.query.topic">Topic Selection</a>
</h2>
<ol node_id="11">
<li>
<a href="#cnn.query.news"> News </a>
</li>
<li>
<a href="#cnn.query.business"> Business </a>
</li>
<li>
<a href="#cnn.query.sports">
          Sports
         </a>
</li>
<li>
<a href="#cnn.query.travel"> Travel </a>
</li>
<li>
<a href="#cnn.query.weather"> Weather </a>
</li>
<li>
<a href="#cnn.query.show">
           Show Business
         </a>
</li>
</ol>
    <h2>
<a name="cnn.query.news">News Channel</a>
</h2>
<form node_id="13" action="http://raman.almaden.ibm.com/cgi-bin/
cnn.cgi">
      <P node_id="9">
      Which part of today's news would you like to read?</P>
<select name="cnn.query.part">
          <option value="h"> Headlines</option>
          <option value="1"> first story </option>
          <option value="2"> second story </option>
          <option value="3"> third story </option>
        </select>
<p>
< /p>
      <P node_id="14">
          Which news category would you like to read?
        </P>
```

```
<select name="cnn.query.interest">
          <option value="business">
            Business
          </option>
          <option value="africa">
            Africa</option>
          <option value="world"> World </option>
          <option value="United states"> United states </option>
          <option value="europe"> Europe </option>
          <option value="Asia"> Asia</option>
          <option value="me"> Middle East</option>
          <option value="america"> America </option>

</select>
<p>
</p>
    <p>
<INPUT TYPE="SUBMIT" VALUE="cnn.query.news">
< /p>
</form>
    <h2>
<a name="cnn.query.business">Business Channel</a>
</h2>
<form node_id="16" action="http://raman.almaden.ibm.com/cgi-bin/
cnn.cgi">
      <P node_id="9">
Which part of today's news would you like to read?</P>
<select name="cnn.query.part">
          <option value="h"> Headlines</option>
          <option value="1"> first story </option>
          <option value="2"> second story </option>
          <option value="3"> third story </option>
        </select>
<p>
</p>
      <P node_id="17">
Which business category would you like to read?</P>
<select name="cnn.query.interest">
          <option value="NEWS"> news </option>
          <option value="IN"> indexes </option>
          <option value="CU"> exchange rates </option>
          <option value="MET"> metals </option>
        </select>
<p>
</p>
    <p>
<INPUT TYPE="SUBMIT" VALUE="cnn.query.business">
< /p>
</form>
    <h2>
```

```
<a name="cnn.query.weather">Weather Channel</a>
</h2>
<form node_id="19" action="http://raman.almaden.ibm.com/cgi-bin/
cnn.cgi">
      <P node_id="9">
Which part of today's news would you like to read?</P>
<select name="cnn.query.part">
          <option value="h"> Headlines</option>
          <option value="1"> first story </option>
          <option value="2"> second story </option>
          <option value="3"> third story </option>
        </select>
<p>
</p>
      <P node_id="20">
          Which region are you interested in?</P>
<select name="cnn.query.interest">
          <option value="us"> United states </option>
          <option value="europe">

          Europe
          </option>
          <option value="JP"> Japan </option>
          <option value="AU"> Australia </option>
          <option value="AS"> Asia </option>
        </select>
<p>
</p>
    <p>
<INPUT TYPE="SUBMIT" VALUE="cnn.query.weather">
</p>
</form>
    <h2>
<a name="cnn.query.travel">Travel Section</a>
</h2>
<form node_id="22" action="http://raman.almaden.ibm.com/cgi-bin/
cnn.cgi">
      <P node_id="9">
          Which part of today's news would you like to read?</P> <select
 name="cnn.query.part">
              <option value="h"> Headlines</option>
              <option value="1"> first story </option>
              <option value="2"> second story </option>
              <option value="3"> third story </option>
            </select>
<p>
</p>
      <P node_id="23">
          Which city do you want to visit?</P>
<select name="cnn.query.interest">
```

```
          <option value="AMSTERDAM">AMSTERDAM</option>
          <option value="COPENHAGEN">COPENHAGEN</option>
          <option value="HELSINKI">HELSINKI</option>
          <option value="HONGKONG">HONGKONG</option>
          <option value="LONDON">LONDON</option>
          <option value="OSLO">OSLO</option>
          <option value="PRAGUE">PRAGUE</option>
          <option value="SINGAPORE">SINGAPORE</option>
          <option value="STOCKHOLM">STOCKHOLM</option>
          <option value="SYDNEY">SYDNEY</option>
        </select>
<p>
</p>
<p>
<INPUT TYPE="SUBMIT" VALUE="cnn.query.travel">
</p>
</form>
    <h2>
<a name="cnn.query.sports">Sports Channel</a>
</h2>
<form node_id="25" action="http://raman.almaden.ibm.com/cgi-bin/
cnn.cgi">
      <P node_id="9">
      Which part of today's news would you like to read?</P>
<select name="cnn.query.part">
          <option value="h"> Headlines</option>
          <option value="1"> first story </option>
          <option value="2"> second story </option>
          <option value="3"> third story </option>
        </select>
<p>
</p>
      <P node_id="26">
          What sports are you interested in?</P>
<select name="cnn.query.interest">
          <option value="AS"> Asia </option>
          <option value="w"> world </option>
          <option value="eu"> europe </option>
          <option value="us"> united states </option>
          <option value="nba"> NBA </ option>
          <option value="nhl"> nhl </option>
          <option value="EF"> Europoean football </option>
        </select>
<p>
</p>
    <p>
<INPUT TYPE="SUBMIT" VALUE=" cnn.query.sports">
</p>
</form>
  </div>
```

```
</div>
</body>
</html>
```

【０１５６】Ｎ．ＣＭＬ　ＤＴＤ　文書型定義

以下は、ＣＭＬ　ＤＴＤを表現する。以下のＤＴＤ記述が、ＸＭＬの技術に精通する者なら誰にでも完全に理解されるべきであることを理解されたい。これは、この実施形態に対して呈示されるＣＭＬの構文を完全に定義する。

```
<!--$Id: cml.dtd,v 1.14 2000/03/02 17:04:02$ -->
<!-- DTD For Conversational Markup Language CML -->
<!--Conventions:
Tags are all lower case.
Attribute names are all lower case. -->
<!-- {attribute entities -->
<!-- core attributes common to most elements
node_id document-wide unique id
name Names data item that is populated by this gesture.
title Human readable title
style URI of custom stylesheet
-->
<!ENTITY % coreattrs
"node_id ID #IMPLIED
name CDATA #IMPLIED
style CDATA ; #IMPLIED
trigger CDATA #implied
title CDATA #IMPLIED"
>
<!-- } -->
<!-- {entities -->
<!ENTITY % GESTURE "(cml
| select
| menu
| message
| help)">
<!-- } -->
<!-- { TOP LEVEL CML -->
<!ELEMENT group (
%GESTURE+)
>
<!ATTLIST group
id ID #required
modality CDATA #implied
class CDATA #implied
>
<!ELEMENT CML (
(group | %GESTURE)+,
submit?
)
>
<!ATTLIST cml %coreattr>
<!-- } -->
<!-- {gesture message
```

```
<!ELEMENT message ANY>
<!ATTLIST message %coreattr>
<!-- } -->
<!-- {gesture help
<!ELEMENT help ANY>
<!ATTLIST help %coreattr>
<!-- } -->
<!-- {gesture boolean
<!ELEMENT boolean(
message,
help?)
>
<!ATTLIST boolean %coreattr;
require_confirmation (true | false ) #implied
require_confirmation_if_yes (true | false #implied
require_confirmation_if_no (true | false #implied
default (true | false #implied
>
<!-- } -->
<!-- {gesture select
<!ELEMENT error ANY>
<!ELEMENT grammar (
gram,
help?)
>
<!ATTLIST grammar
type CDATA #required>
<!ELEMENT gram ANY>
<!ELEMENT final ANY>
<!-- open content model for element predicate for now -->
<!-- will use an expression syntax a la xpath and augmented -->
<!-- as needed -->
<!-- will also draw on xforms work -->
<!ELEMENT predicate ANY>
<!ELEMENT choice (
grammar?,
PCDATA)
>
<!ATTLIST choice %coreattr;
value CDATA #required
>
<!-- default has same content model as choice -->
<!ELEMENT default (
grammar?,
PCDATA)
>
<!ATTLIST default %coreattr;
value CDATA #required
>
<!ELEMENT choices (
```

```
choice+,
default?)
>
<!ELEMENT select (
message,
help?,
choices,
predicate?,
error?)
>
<!ATTLIST select %coreattr;
require_predicate (true | false ) #implied
selection_type CDATA #implied
>
<!-- } -->
<!-- {gesture menu
<!ELEMENT menu (
message,
help?,
choices)
>
<!ATTLIST menu %coreattr; >
<!-- } -->
<!-- {constrained input -->
<!-- CML provides gestures for standard dialog components,
the following is merely a sample list of gestures:
Date
Specify date
Time
Specify time.
Currency
Specify currency amount.
Credit card
Specify a credit card (including card type, card number and
expiration date).
Phone
Specify a telephone number.
Email
Specify an email address.
url
Specify a url.
Snail Adress
Specify a snail mail address, including street, city/state/country
and zip code.
We will specify formal DTD for these elements. -->
<!-- } -->
<!-- {unconstrained input -->
<!ELEMENT input (
message,
help?,
```

```
predicate?)
>
<!ATTLIST input %coreattr;
require_predicate (true | false ) #implied
>
<!-- } -->
<!-- {gesture user_identification
<!ELEMENT user_identification (
message,
help?,
user,
identify,
predicate,
error)
>
<!ATTLIST user_identification %coreattr;
require_predicate (true | false ) #implied
on_fail CDATA #implied
>
<!-- } -->
<!-- {gesture submit -->
<!ELEMENT env EMPTY>
<!ATTLIST env
name CDATA #required>
<!ELEMENT submit (
message?,
help?,
env*)
>
<!ATTLIST submit
target CDATA #required>
<!-- } -->
<!-- {binding events -->
<!ELEMENT bind-event EMPTY>
<!ATTLIST bind-event
logical CDATA #required
physical CDATA #implied
modality CDATA #implied
>
<!-- } -->
<!-- {environment
<!ELEMENT var EMPTY>
<!ATTLIST var
name CDATA #required
value CDATA #implied
>
<!ELEMENT value EMPTY>
<!ATTLIST var
name CDATA #required
>
```

```
<!ELEMENT assign EMPTY>
<!ATTLIST var
name CDATA #required
value CDATA #required
>
<!-- } -->
<!-- { end of file -->
<!-- End Of DTD
local variables:
folded-file: t
end:
-->
<!-- } -->
```

【０１５７】したがって、本発明による会話型マークアップ言語は、本明細書に詳細に述べるように、多くの有用な特徴および利点を提供する。対話によるプログラミングは、実装されている、基礎をなすデータ・モデルの定義（モデル）が、ユーザ対話を定義するマークアップ言語（ビュー／コントローラ）から分離されることを可能にする。これにより、密に同期化されたマルチモーダル対話の構築が可能となり、会話型アプリケーションがサポートされる。本発明によるＣＭＬは、密な同期化をサポートする機構、すなわち各ジェスチャにNode_id属性を結合し、この属性を様々な出力にマッピングすることを提供する。この言語は、原子構成（ジェスチャ）で定義されることが好ましく、より複雑な構成がもしあれば、それらは、（ダイアログで表した）複雑な構成の明瞭な意味定義と共にこれらの原子ジェスチャで構成される。これにより、複雑なモジュールを異なるモーダル性にマッピングすることができる。音声は、ＧＵＩと同じレベルにある第１のクラスのユーザ・インタフェース（ＵＩ）モーダル性と考えられる。ジェスチャは、基本のダイアログ・コンポーネントに対応する（これは、適切なデータ・ファイルを追加することを含む）。必要とされる場合に、モーダル性特定のコンポーネントをカプセル化したいと思う作成者は、モーダル性特定のマークアップをコード化するための「通過」機構を提供することができる。モーダル性特定の構造（スピーチ用またはＧＵＩ用）は、この通過機構に限定される。会話型ＵＩがサポートされる。マークアップ言語は、並列で活動化できるダイアログ・コンポーネントを取り込む。ＣＭＬは拡張可能言語であり、例えば、新しいジェスチャを定義することができ、ジェスチャ変換規則を修正することができ、他の言語からのタグ／構成を埋め込むことができる（通過モードで）。モーダル性特定のタグ／通過は、ページを追加で表面変更するための機構でしかない。ＣＭＬはまた、アプリケーション状態をカプセル化するための明示的な環境も提供する。ＣＭＬはまた、動的に生成されたデータに言及する対話記述ならびにバックエンドへのサポート用コールバック機構の能力も提供する。どんな従来方法も、これらの目的に使用することができる。さらに、本明細書に提供するＣＭＬの詳細な説明が与えられれば、当業者は、この創意に富んだマークアップ言語の使用に関連する様々なツールおよび開発環境を実現することができる。

【０１５８】ＩＩ．マルチモーダル・ブラウザ

以下は、本発明によるマルチモーダル・ブラウザの説明である。このセクションは、参照しやすいように次のサブセクションに分かれている。すなわち、（Ａ）序説、（Ｂ）マルチモーダル・シェル、（Ｃ）マルチモーダル・シェルとＣＭＬ、（Ｄ）ＣＭＬとマルチモーダル同期化、（Ｅ）ＣＭＬとアプリケーション・オーサリング、（Ｆ）例示的な実施形態、（Ｇ）代替実施形態である。

【０１５９】Ａ．序説

本発明によるマルチモーダル・ブラウジングを説明する前に、本発明のＣＭＬおよびマルチモーダル・ブラウザに関係する概念を有する、上に参照した特許出願のいくつかの記述を要約したものを以下に述べる。参照しやすいように、関連出願は、それらの各整理番号を介して言及する。

【０１６０】ＹＯ９９９－１１１は、会話型コンピューティング、会話型ユーザ・インタフェース、および会話型アプリケーション・プラットフォーム（ＣＶＭ、会話型仮想マシン）の概念を開示する。ＹＯ９９９－１１１に述べられている、ＣＶＭから提供される機能および挙動／サービスは、実際には、本発明のマルチモーダル・ブラウザによって、または会話型ユーザ・インタフェースを提供するアプリケーションによって実施される。しかし、概念レベルでは、ＣＶＭが、本発明のブラウザをサポートするのに必要なすべてのサービスを実施すると仮定される。

【０１６１】ＹＯ９９８－３９２は、会話型アプリケーション（すなわちマルチモーダル）をプログラムするための宣言的なプログラミング言語（「ＣＭＬ」と言及されるが、これはこの場合、本発明の言語とは異なる）の使用を開示する。ＹＯ９９８－３９２言語は、マルチモーダル／会話型ユーザ・インタフェースをサポートする

宣言的な言語である。実際には、その明細書に提供されている例／実施形態は、本発明によって可能な単一のオーサリングではなく「複数オーサリング」モデルに従って書かれたＭＬページからなる。以下の通り、宣言的なプログラミング言語の異なる例が教示された。

（ｉ）スピーチのみのＭＬ。スピーチＭＬとも呼ばれ、VoiceＸＭＬに通じる。

（ii）ファイル間の同期化タグを有する複数のファイル（ＨＴＭＬとVoiceＭＸＬ、またはＷＭＬとVoiceＸＭＬ）。

（iii）複数のモーダル性記述を有する単一のファイル（例えば<MM><Speech>Speech rendering info</speech><GUI>GUI rendering info</GUI></MM>など）。この場合も同期化情報を有する。

（iv）異なるモーダル性に関連する情報を分割するためのフレーム様モデルを有する単一のファイル（例えば、スピーチ・コンテンツがＨＴＭＬページに加えて「スピーチ・フレーム」中に呈示される）。

【０１６２】これらのアイテムはどれも、単一オーサリングに対処しない。また、ＣＭＬ、任意のターゲット・レガシーＭＬ（チャネル）、あるいはジェスチャまたはジェスチャ・ベースのＸＳＬの概念からのサポートにも対処しない。

【０１６３】ＹＯ９９９－１７８は、一般マルチモーダル・シェルを記述する。これは、同期化されるマルチモーダル・アプリケーション（宣言型でも命令型でもハイブリッドでも）をどのようにサポートおよびプログラムするかを記述する。これは、各アプリケーション・モーダル性がその状態、それがサポートするコマンド、およびそれらのコマンドが他のモーダル性に与える影響を登録する登録テーブルを使用する。この場合もまた、ジェスチャおよび単一オーサリングの教示はない。一実施形態は、アプリケーションがブラウザ（すなわち各モーダル性のレンダリングに関連するブラウザ）であり、シェルがＣＭＬページを受け取り（ＹＯ９９８－３９２に定義されるように）、登録テーブルを構築し、したがってモーダル性にわたって同期をとる体系を記述する。

【０１６４】ここで本発明は、以下の記述に説明するように、マルチモーダル・ブラウザ体系を可能にする。このようなマルチモーダル・ブラウザは、以下に述べるが、上のセクションＩで詳細に述べたＣＭＬおよびその言語の会話型ジェスチャの特徴および利点を利用して、ユーザがアプリケーションによってサポートされる任意の装置上にある任意のモーダル性の情報にアクセスできるようにする。例えば、マルチモーダル・ブラウザ内の視覚的および話される対話は、会話型ジェスチャのコア・セットを使用して抽象され、ＣＭＬを使用して表現される。対話型ジェスチャは、各対話モーダル性によって適切に実現される。これらの基本的な会話型ジェスチャを使用して軽量情報アプリケーション（インフォウェア）をオーサリングすることができ、得られたコンテンツは、表現されるときに、モーダル性／装置特定のマークアップ言語またはワイヤ・プロトコル（wire protocol）、例えばいくつかを挙げればVoiceＸＭＬ、ＷＭＬに投影される。

【０１６５】Ｂ．マルチモーダル・シェル

マルチモーダル・ブラウザの動作の中心には、マルチモーダル・シェル機構がある。マルチモーダル・シェルは、複数のユーザ・インタフェース・クライアントまたはブラウザに対するサーバの役割を果たす。異なる対話モーダル性を備えるブラウザ、例えば視覚ＨＴＭＬブラウザまたは聴覚VoiceＸＭＬブラウザは、マルチモーダル・シェルにクライアントとして登録する。ユーザ対話は、マルチモーダル・シェルがＣＭＬ文書を横断することによって進行する。この横断中に、シェルは、以下によって、ユーザの対話を特定の個々のＣＭＬインフォウェアに対して調整する。

（ｉ）現行のＣＭＬノードの対話特定の表現を、すべての登録済みクライアントに渡すことによってユーザ対話を開始する。

（ii）現行のＣＭＬノードを受け取ったすべての登録済みクライアントからの情報更新を待機する。

（iii）受け取った情報間の矛盾、例えばユーザが右と言っているのに左を指しているなどをおそらく解消する。

（iv）受け取ったばかりの情報更新に基づいて現行のＣＭＬノードを更新する。

（ｖ）更新の実行が成功すると、シェルは新たに更新されたアプリケーション状態をすべての登録済みブラウザに渡す。

【０１６６】Ｃ．マルチモーダル・シェルとＣＭＬ

上に説明したように、ＣＭＬアプリケーションは、標準的な会話型ジェスチャのセットの集合である。このような会話型ジェスチャは、アプリケーションを構成する完全なダイアログの基本的な構成単位を形成する。例えば、特定のアプリケーションでは、アプリケーション設計者の主要な作業は、以下のものを指定することである。

（ｉ）ユーザから収集する情報のアイテムを指定する。

（ii）必要なアイテムごとに制約、例えばセットからの選択などを指定する。

（iii）情報の各アイテムに必要なものが備わったときに、アプリケーション状態を更新する。

（iv）収集した情報のアイテムをパッケージして、バックエンド・アプリケーション・サーバにサブミットする。

【０１６７】上の作業は、指定されるとき、使用中の対話モーダル性に対して独立であることに留意されたい。

【０１６８】異なるユーザ・インタフェース・フロントエンド、例えば視覚ＷＷＷブラウザ、聴覚VoiceＸＭＬ

などは、これらの作業を適切なユーザ・インタフェースの仕掛けにマッピングする。

【０１６９】ＣＭＬ文書は、一般マルチモーダル・シェルによってホスティングされる。シェルは、異なるユーザ・インタフェースの実現、例えば、視覚ＨＴＭＬブラウザまたは聴覚VoiceＸＭＬブラウザを助ける。シェルのクライアントになりたいと思うブラウザは、現行のアプリケーション状態への弱い参照を保持する。登録済みクライアントは、アプリケーション状態が変化したときにシェルから通知され、次いで各クライアントは、それ自体の弱い参照をアプリケーション状態に照会して、ユーザに呈示したいと思う関係情報を抽出する。

【０１７０】ユーザは、登録済みブラウザの１つを介してアプリケーションと対話することによって、ＣＭＬ文書を横断する。対話が進行するにつれて、すべての登録されたブラウザは、対話のフォーカスである現行のＣＭＬノードについて通知され、したがって、それらの表現を必要に応じて更新する。シェルは、現在開かれているＣＭＬ文書、ならびにそれらに対応するアプリケーション状態を常に監視する。必要な場合に、会話型シェルは、現在開かれているアプリケーションのいずれかの状態の簡潔な要約を提供することができる。登録済みクライアントのいずれかを介してサブミットされた情報はシェルによって媒介され、シェルは、他の登録済みクライアントへの、かつ必要な場合にバックエンド・アプリケーション・サーバへの通知を扱う。

【０１７１】Ｄ．ＣＭＬおよびマルチモーダル同期化

単一のＣＭＬ表現からのアプリケーションを対話特定に具現化したもの総合することによって、マルチモーダル・インタフェースの異なる態様を同期させることができる。ＣＭＬ表現中の各ノードは、特定のnode-idでタグ付けされる。ＣＭＬ表現が対話特定の表現、すなわちＨＴＭＬまたはVoiceＸＭＬにマッピングされるとき、得られたマッピングにおけるノードは、それらに対応するＣＭＬ表現中のノードのnode-idでタグ付けされる。ユーザが特定のモーダル性を介してブラウザと対話するとき、マルチモーダル・シェルは、関係するnode-idを検索することによって、アプリケーション中の現在活動化されているノードを元のＣＭＬ表現にマッピングし返す。ユーザ対話のせいでアプリケーション状態が変化するとき、シェルは、修正されたアプリケーション状態を、修正されたノードのnode-idと共に、通知されるように登録されたすべてのクライアントに渡す。通知されたアプリケーションは、node-idに対してチェックすることによって、それらの対話特定の表現中の対応するノードを更新する。登録済みのアプリケーションは本質的に、基礎をなすアプリケーション状態への弱い参照を保持することに留意されたい。対話特定のレンダリング・エンジンが必要なノードを更新するとき、弱い参照は、その更新に関係する情報（かつ、必要とされる情報だけ）が自動的にシェルから検索されるようにすることになる。

【０１７２】ここで図１２を参照すると、ＭＶＣモデルの新しい解釈が示されている。この新しい解釈によれば、このモデルは対話のＣＭＬ記述である。ビューは、ジェスチャ・ベースのＸＳＬ変換規則を適用して、異なるレンダリング・ブラウザ中にレンダリングされる（ビュー）異なるターゲットＭＬを生成することによって得られるものである。ブラウザは、ユーザとの対話を通してモデルの制御を提供する（かつ、Ｉ／Ｏイベントがレンダリング・ブラウザの１つで発生したときにその状態を修正する）。図９に従って、Ｖ０がＧＵＩビュー（例えばＨＴＭＬ）でありＶ１がスピーチ・ビュー（自然言語による、またはそうでない）であると想像されたい。Ｃ０は、モノモーダルＨＴＭＬブラウザのみの制御／対話である。Ｃ１は、同期化されるマルチモーダル・ビューである。Ｃ２は、モノモーダル・スピーチ制御である。この手法は、根本的に新しいパラダイムである。

【０１７３】Ｅ．ＣＭＬとアプリケーション・オーサリング

アプリケーション・クリエータは、ＷＹＳＩＷＹＧ（what you see is what you get）オーサリング・ツールと対話して、自分のアプリケーションのＣＭＬ表現を作成することができる。ＣＭＬで表現されたアプリケーションは、スタイル変換の標準セットを使用して、対話特定の表現、例えばVoiceＸＭＬまたはＨＴＭＬにマッピングされる。必要とされる場合に、ユーザ・インタフェース設計者は、カスタム・スタイル変換を生み出して、特定のルック・アンド・フィールまたはサウンド・アンド・フィールを設計することができる。ＣＭＬオーサリング・ツールを生み出すこともでき、これは、クライアントがマルチモーダル・ブラウザ・プラットフォーム上で展開するためにレガシーＨＴＭＬのみのWWWアプリケーションをＣＭＬにマッピングすることができるように作成することができる。このようなツールは、顧客が既存のWWWアプリケーションをVoiceＸＭＬプラットフォーム上で展開することを助けるのに必要な橋を提供する。この解決法は、VoiceＸＭＬに直接オーサリングし直すよりも魅力がある。というのは、既存のアプリケーションをＣＭＬに一度マッピングすることは、様々なマルチモーダル・ブラウザ設定にわたる展開を可能にするからである。これは、ＨＴＭＬ、ＷＭＬ（およびその他のレガシーＭＬ）にも当てはまる。

【０１７４】Ｆ．例示的な実施形態

次に図１３～１５を参照すると、既存のシステムから本発明によってマルチモーダル・ブラウジング環境でＣＭＬを完全に使用するまでの移行ロード・マップが示されている。

【０１７５】図１３に、現行のファット・クライアント・ウェブ・プログラミング・モデルを示す。コンテンツ

は、主にＨＴＭＬで書かれる（そのフォーマットで静的に記憶される、または動的に生成される）。コンテンツが特定のブラウザ（例えば所与のバージョンのInternet explorerまたはCommunicator）に適合される必要があるとき、ターゲット・ブラウザの機能である特定のスタイル・シート、ならびにコンテンツのタイプが構築される。これは通常、ＸＭＬ／ＸＳＬオーサリング手法である。別のチャネル／モーダル性（ＷＭＬ、ＣＨＴＭＬ、VoiceＸＭＬなど）が必要な場合、コンテンツは、書直す必要があるか、あるいはＨＴＭＬまたはＸＭＬで書かれた場合には非常に特定の規則に従って、よく知られたタイプ／領域のものとする必要があり、したがって、いくつかの一般アプリケーション／ビジネス論理に依存するＸＳＬ規則を使用してこれらのモーダル性特定のレガシー言語を作成することができるか、またはＸＳＬ規則を非常に頻繁にオーサリングし直さなければならないか、あるいはその両方である。これは、異なるレガシー言語で直接にオーサリングしようが、単一のＸＭＬコンテンツをこれらの異なるレガシーＭＬに変換する異なるスタイル・シートでオーサリングしようが、過多な複数オーサリングとなる。最終的に、今日、ウェブ（すなわち主にＨＴＭＬの交換による）、無線ネットワーク（主にＷＭＬ、ただし他の規格も存在する）、および電話（主にVoiceＸＭＬ）にアクセスする必要が一層多くなっている。複数オーサリングが唯一の解決法のため、このようなタイプのサービスを提供するサイトは、通常、限られた量のサービス／コンテンツ・プロバイダまたは企業サイトによる、閉じたサイト（開いた完全なウェブ・コンテンツとは反対に、限られた量のサービス／コンテンツ）である。いつどこでも、どんな情報へのアクセスをどんなアクセス装置を介しても提供し、ユーザがそれを操作できるようにする解決法は存在しない。異なるレガシー言語（ＸＭＬを含む）は、ページの異なる部分を適切に他のモーダル性で扱うのに必要な情報を含まない（例えば会話エンジン用の文法およびその他の引数が欠落しているなど）。

【０１７６】図１４に、ＣＭＬを展開して、対話プログラミング・モデルおよび会話型コンピューティング・パラダイムによるプログラミングを使用する第１のステップを記述する。この解決法は、トランスポート・プロトコルおよびネットワーク（例えばテレフォニーＰＳＴＮ、無線ネットワーク（音声またはデータあるいはその両方）、voice over ＩＰ、ＴＣＰ／ＩＰ－ＨＴＴＰ、ＷＡＰなど）、ならびにレガシー・ブラウザ（例えばＨＴＭＬブラウザ、ＷＭＬブラウザ、VoiceＸＭＬブラウザなど）で表される、今日存在するインフラストラクチャを使用することができる。コンテンツがＣＭＬで利用可能な場合、それは、静的に生成されようが動的に生成されようがページが提供されるとき、要求するブラウザによってサポートされるターゲット・レガシーＭＬに実行中にトランスコーディングすることができる。ターゲットＭＬの決定は、ブラウザのタイプ、またはゲートウェイ、ブラウザ、サーバのＩＰに基づく。ＷＡＰゲートウェイはＷＭＬページを受け取り、ブラウザは、（http ヘッダ中の）記述子またはアクセス機構に基づいてその要件を記述する（例えば、httpは、いくつかのＣＭＬブラウザが利用可能になるまでの少なくとも展開の最初では、ＨＴＭＬを暗黙指定することになる）。決定はまた、要求されたページに基づいても行われる。ブラウザがxxxx.htmlを要求する場合、それは、ＣＭＬがＨＴＭＬにトランスコーディングされることを意味し、yyyy.vxmlを要求する場合、それは、VoiceＸＭＬにトランスコーディングされることを意味する、などである。明らかにこれは、現在のインフラストラクチャおよびその何らかの未来の進化形をサポートすることを保証する。

【０１７７】ＣＭＬブラウザ（すなわち会話型／マルチモーダル）が開放されるとき、それは、ＣＭＬページ（すなわちzzzz.cml）を要求することになり、また、それ自体をＣＭＬブラウザとして記述することもできる。このような場合、ページはどんなトランスコーディングも必要とせずに提供される。これは、レガシー／今日のインフラストラクチャからＣＭＬ／会話型支配のウェブ・プログラミング・パラダイムへのスムーズな移行を保証する。ここで、レガシー・コンテンツ（すなわちＨＴＭＬ、ＷＭＬ、VoiceＸＭＬ、またはその他のレガシー言語、あるいはそれらすべてで書かれた静的または動的コンテンツ）は、ＣＭＬに変換される必要がある。ツールは、せいぜいＣＭＬターゲットを「当てる」ことにしか使用できず、次いでそのＣＭＬターゲットは、手作業で妥当性検査して再編集する必要がある。しかし、上に説明したのと同じ理由で、元のページが特定の規則に従って構築された場合、またはＸＭＬタグが明確であり（領域特定であり）、したがってページ中のそれらの役割が明確である場合、実行可能な自動トランスコーディング・システムを使用することができる。

【０１７８】図１５に、ＣＭＬ会話型（マルチモーダル・ブラウザ）が標準となったときの展開ロード・マップにおける次のステップを示す。したがって、今やトランスコーディングはブラウザの一部であり、ページはＣＭＬでオーサリングされ提供される。レガシー（すなわち非ＣＭＬ）ページが提供されるとき、それは、マルチモーダル・シェルによって取り出されるが、次いで、対応するモーダル性を扱う対応するレンダリング・ブラウザに直接に送信されることになる。

【０１７９】ＣＭＬコンテンツおよびレガシー・コンテンツは、当然、依然として前述のようにＣＭＬにオーサリングまたは変換される必要がある。

【０１８０】図１６を参照すると、本発明によるマルチモーダル・ブラウザ体系のブロック図が示されている。図示のように、マルチモーダル・ブラウザ６０は、マル

チモーダル・シェルまたは会話型シェル６２、ＧＵＩレンダリング・ブラウザ・コンポーネント６４、およびスピーチ・レンダリング・ブラウザ・コンポーネント６６を備える。マルチモーダル・シェルはまた、「仮想ブラウザ」とも呼ばれる。マルチモーダル・ブラウザ６０は２つのモーダル性、視覚（ブラウザ・コンポーネント６４）およびスピーチ（ブラウザ・コンポーネント６６）の使用を示しているが、本発明がこれらのモーダル性に限定されないことを理解されたい。マルチモーダル・ブラウザ６０は、一般に次のように動作する。アプリケーションにアクセスしようとするユーザは、マルチモーダル・ブラウザのすべてか一部が常駐するクライアント装置（例えばパーソナル・コンピュータ、ラップトップ・コンピュータ、携帯情報端末など）とインタフェースする。図１６に示す一般的な場合では、ユーザはテキスト・インタフェースもしくはグラフィカル・インタフェース（ＧＵＩ入力／出力）またはその両方を介してこれを行うことができ、あるいはインタフェースはスピーチ（オーディオ入力／出力）を介したものとすることができ、あるいはその両方ができる。図１６にはマルチモーダル・ブラウザ６０を１つのブロック中に示してあるが、マルチモーダル・ブラウザはクライアントとサーバのコンピュータ・システムを両方含めた複数の装置を介して実施できることを以下に説明する。

【０１８１】ユーザの要求に基づいて、マルチモーダル・ブラウザ６０は適切なＵＲＬをコンテンツ・サーバ６９に送り、コンテンツ・サーバ６９もまた、所望の特定アプリケーションへのアクセスを要求するために、同じくクライアント装置に常駐することのできる会話型エンジン６８にサービスする。次いで、アプリケーションに関連するＣＭＬコードが、コンテンツ・サーバ６９からマルチモーダル・ブラウザ６０にダウンロードされる。次いでマルチモーダル・ブラウザは、そのＣＭＬコードに関連する会話型ジェスチャに基づいて、モーダル性特定のレンダリング（ＧＵＩ表現またはスピーチ表現あるいはその両方）を生成する。したがってユーザは、これらの表現を介してブラウザ６０と対話する。

【０１８２】次に（引き続き図１６も参照しながら）図１７を参照すると、本発明の一実施形態によるマルチモーダル・ブラウザの動作をより詳細に示す流れ図が示されている。アプリケーション開発者は、アプリケーション、例えばインフォウェアと呼ばれる軽量アプリケーションをＣＭＬで書く。ＣＭＬでオーサリングされたインフォウェアは、複数のモーダル性特定のブラウザ・コンポーネント（例えば図１６の視覚ブラウザ６４とスピーチ・ブラウザ６６）の間を媒介する会話型シェル（例えば図１６のマルチモーダル・シェル６２）によってホスティングされる。マルチモーダル・シェルは、ＣＭＬインタープリタまたはプロセッサと考えることができる。これを、ブロック７０として図１７に示す。ユーザ対話はＣＭＬインタープリタによって進行し、ＣＭＬインタープリタは、ダウンロードされたＣＭＬコードに関連するＣＭＬインスタンスをＨＴＭＬ（ブロック７７）やVoiceＸＭＬ（ブロック７８）などの適切なモーダル性特定の言語にマッピングする。これらのモーダル性特定の表現は、そのアプリケーションに関連するモーダル性特定のバージョンのダイアログをレンダリングする。ブロック７０中に示すように、ノード（Ａ）および矢印（Ｂ）は、ＣＭＬの宣言型プログラムを表す。ＣＭＬプログラム中のジェスチャは各ノードで表され、矢印は、あり得る分岐点またはループによる対話／ダイアログの流れを表す。各ジェスチャはノードＩＤ（node_id）で識別され、これは、異なる登録済みのモーダル性の間で同期をとるために、活動化されたジェスチャを適切に識別することを可能にする。node_idはジェスチャを識別し、したがって、ＣＭＬブラウザ（すなわちマルチモーダル・シェルまたは仮想ブラウザ）は、それがダイアログ・フロー中のどこであるか、およびそこからどこに行くべきか（例えば、異なるモーダル性を更新する、またはサーバに変数を送って新しいＣＭＬページを取り出す）を知る。

【０１８３】ＣＭＬからモーダル性特定の表現への変換７７および７８は、ＸＳＬ変換規則（または前述の他の変換機構）によって支配される。これらのＸＳＬ規則は、モーダル性特定である。これらの変換は、ＸＳＬ規則７４および登録テーブル７６に従って、プレゼンテーション生成ブロック７２によって扱われる。登録テーブル７６は、デフォルトのジェスチャＸＳＬ変換規則、ならびに拡張形、アプリケーション特定、装置特定、またはユーザ特定の特定規則のリポジトリである。ＣＭＬインスタンスを適切なモーダル性特定の表現にマッピングする過程で、ＸＳＬ規則は、モーダル性特定のユーザ対話を実現するのに必要な情報を追加する。一例としては、要素selectをVoiceＸＭＬに変換するとき、関係するＸＳＬ変換規則は、その会話型ジェスチャに有効な選択肢をカバーする文法の生成を扱う。

【０１８４】ＣＭＬインスタンスをＨＴＭＬなどのモーダル性特定の表現に変換する過程は、単一のＣＭＬノードを出力表現中のノードの集合にマッピングすることになる。これらの様々な表現にわたって同期をとるのを助けるために、ＣＭＬ属性node_idが、所与のＣＭＬノードから得られた出力ノードのすべてに適用される。所与のＣＭＬインスタンスが適切なモーダル性特定のＸＳＬ規則によって異なる表現、例えばＨＴＭＬおよびVoiceＸＭＬにマッピングされるとき、出力におけるツリーの形は、様々なモーダル性の間で異なる見込みが高い。しかし、属性node_idは、モーダル性特定の各表現から元のＣＭＬノードへの概念上のバックリンクを提供することにより、これらの表現の間で同期をとることを可能にする。これを、図１７のブロック７０に図示する。

【０１８５】ユーザ対話が進行するにつれて、現行のＣＭＬインスタンスによって環境中で定義される変数は、妥当性検査された値に束縛される。この束縛は、まず、モーダル性特定の表現（登録済みクライアント）７７および７８のうちの１つで発生する。モーダル性特定の表現は、更新された環境および完了したばかりのジェスチャのnode_idを含む適切なメッセージをＣＭＬインタープリタ（マルチモーダル・シェル）に送る。更新された束縛がＣＭＬインタープリタに伝達されると、ＣＭＬインタープリタは、完了したばかりのジェスチャのnode_idでモーダル性特定の表現すべてにメッセージを送る。モーダル性特定の表現は、このメッセージを受け取ると、まずＣＭＬインタープリタにそれらの表現に影響する環境の部分を照会することによって、それらの表現を更新する。

【０１８６】図１８に、本発明の一実施形態によりＣＭＬマルチモーダル・ブラウザによって行われる異なるステップを示す。ＣＭＬページがブラウザによって取り出されると、ブラウザは、ＸＭＬパーサと同様にＣＭＬコンテンツを解析する（ステップ９０）。ブラウザは、対話の内部表現（すなわち、ページ中に記述された異なるジェスチャのグラフ／ツリー）およびノードＩＤを構築する。ブラウザは、ブラウザに記憶されたジェスチャＸＳＬ変換（またはJava BeansやJava Server Pagesのような他の変換機構）を使用して（ブロック９８）、異なるＭＬページを構築し（ステップ９６）、これは各レンダリング・ブラウザに送られる（ステップ１００）。あるモーダル性のＩ／Ｏイベント時に、その影響が対話グラフ（すなわちＹＯ９９９－１７８に記載のようにＭＭシェル登録テーブル（ブロック９４）に記憶された）のレベルで検査される（ステップ９２）。ジェスチャＸＳＬ変換規則は、アプリケーション開発者が上書きし、それらをどこでダウンロードすべきかを示すことができることに留意されたい。これらはまた、普通ならデフォルト挙動となるものから、ユーザ、アプリケーション、または装置のプリファレンスによって上書きすることもできる。新しいジェスチャも追加することができ、その場合、関連するＸＳＬ規則が提供されなければならない（例えばそれを得るためのＵＲＬ）。

【０１８７】前述のように、本発明は、複数装置ブラウジング環境または分散ブラウジング環境を可能にする。ＣＭＬの性質およびそれが効果的に複数のブラウザを同期させる能力のせいで、アプリケーションの様々な部分を別個のコンピューティング装置上に常駐させて実行することができる。次いでユーザは、１つのアプリケーションにアクセスするときに複数の装置、例えばラップトップ・コンピュータおよびセルラー・ホンと同時に対話することができる。これは実際、異なるモーダル性でのブラウジングに限定されない。同じモーダル性（例えばＧＵＩのみ）中でも、同じコンテンツを表現してこの表現をモーダル性にわたって同期化する必要があるとき、例えば１つの装置上で画像を、別の装置上でビデオを、３つ目の装置上でテキストと背景を足したものを表示するときに、同じ原理を使用して、前もってそれがどんな装置かを記述することができる。別の例は、１つの装置上でテキストおよび画像を、別の装置上でアプレットを表示するなどである。もっと多くの例も容易に考えられる。これは、カストマイズされたジェスチャまたはジェスチャＸＳＬ規則の使用を必要とすることになる。あるいは、これは、それを行うための（他のジェスチャおよびデフォルト・レンダリングを有する）別のマークアップを必要とすることになる。

【０１８８】次に図１９を参照すると、このような分散ブラウジング環境が示されている。マルチモーダル・ブラウザ６２、視覚ブラウザ６４、スピーチ・ブラウザ６６、会話型エンジン６８、およびコンテンツ・サーバ６９の機能および動作は、図１６および１７に関して上に述べたものと同じである。しかし、見れば分かるように、コンポーネントは複数のコンピューティング装置上に分散されている。例えば、マルチモーダル・ブラウザ６２はサーバ８０上に常駐し、ビジュアル・ブラウザ６４はクライアント装置８２上に常駐し、スピーチ・ブラウザはサーバ８４上に常駐する。これらのクライアント装置およびサーバ装置は、WWW、ローカル・ネットワーク、または他の何らかの適したネットワークを介して通信することができる。ユーザはクライアント装置８２に対してローカルでよく、サーバ８０と８２とはリモートに位置する。あるいは、すべてのまたはいくつかのコンピューティング装置を一所に配置することもできる。ユーザがクライアント装置８２と直接に対話するので、オーディオ入力／出力機能８６（例えばマイクロホンおよびスピーカ）が装置８２に備わり、これらはサーバ８４にあるスピーチ・ブラウザに接続される。見れば分かるように、ＣＭＬアプリケーションの、同期した同じ動作は、マルチモーダル・ブラウザの様々なコンポーネントが別個のコンピューティング装置上に位置していても達成することができる。

【０１８９】本発明の方法を実施するための前述の各クライアント装置およびサーバは、メモリおよびＩ／Ｏ装置に動作可能に結合されたプロセッサを備えることができることを理解されたい。本明細書で使用する用語「プロセッサ」は、例えばＣＰＵ（中央処理装置）を備えるものなど、どんな処理装置も含むものとすることを理解されたい。本明細書で使用する用語「メモリ」は、例えばＲＡＭ、ＲＯＭ、固定記憶装置（例えばハード・ドライブ）、取外し可能記憶装置（例えばディスケット）、フラッシュ・メモリなど、プロセッサまたはＣＰＵに関連するメモリを含むものとする。さらに、本明細書で使用する用語「入力／出力装置」または「Ｉ／Ｏ装置」は、例えばデータを処理装置に入力するための１つまた

は複数の入力装置、例えばキーボード、マイクロホンなど、および処理装置に関連する結果を呈示するための１つまたは複数の出力装置、例えばＣＲＴディスプレイ、スピーカなど、あるいはその両方を含むものとする。入力／出力装置は、モーダル性特定であり、したがって、他の装置を採用することもできる。また、「プロセッサ」は複数の処理装置を指すことができ、処理装置に関連する様々な要素は他の処理装置によって共用できることも理解されたい。したがって、本明細書に述べる本発明の方法を実行するための命令またはコードを含むソフトウェア・コンポーネントは、１つまたは複数の関連する記憶装置（例えばＲＯＭ、固定または取外し可能メモリ）に記憶することができ、利用される準備ができたときに部分的にまたは全体で（例えばＲＡＭに）ロードしてＣＰＵによって実行することができる。

【０１９０】Ｇ．代替実施形態

本発明の教示から自明に得られる可能な拡張の中には次のものがある。

【０１９１】（ｉ）上に考察した複数装置ブラウジング（ある所与のモーダル性でも）。

【０１９２】（ii）複数地理サポート。いくつかのジェスチャ（例えば電話番号、アドレスなど）を、ローカルのフォーマットならびに言語に適合させることができる。これは、テキスト間変換システムに結合して、異なるＸＳＬ規則を簡単に介して完全に自動的なローカライズ機構（select yes/NoがSelect Oui/Nonになる）を実現することができる。あるいは、このような自動的なトランスコーダがない場合に、システムを開発／ローカライズ用ツールの一部として使用して、ローカライズ／国際化、地理／地域の適合を高速化することもできる。

【０１９３】（iii）会話型ファウンデーション・クラス（Conversational Foundation Class）。会話型ファウンデーション・クラスは、モーダル性独立の、かつ並列でおよび逐次的に実行してより複雑なダイアログを構築できる命令型ダイアログ・コンポーネントとして、ＹＯ９９９－１１１に導入された。これらは、会話型アプリケーション・プラットフォーム（ＣＶＭ、会話型仮想マシン）から提供されるサービスと組み合わせられて、プラットフォームが提供するこれらのファウンデーション・クラスのライブラリにロード／リンクすることによって、命令型会話型（マルチモーダル・アプリケーション）のプログラミングを可能にする。各ＣＶＭプラットフォームがこれを提供するので、アプリケーション開発者はそれらを利用することができ、装置によってサポートされるモーダル性内のレンダリングおよびそれらの同期化を心配しなくてよい。したがって、本明細書に提供したＣＭＬ仕様中に宣言的に定義した各ジェスチャは、逐次的に（１つずつ）または並列で（複数のフォームが一度に活動化されるように複数が活動化される）実行できる命令型の実施形態を（例えばJavaで）有することができる。ＣＦＣでのプログラミングは、対話による命令型のプログラミングに相当する。すなわち、何らかの命令型のジェスチャを使用してそれにリンクし、それをバックエンドにフックし、従来のコードによってそのジェスチャを共に結合する。このコードまたはＣＦＣ引数中に、いくらかのモーダル性特定のカストマイズを加えることもできる。次いで、プラットフォーム（同じレベルの機能を実行するＣＶＭまたはブラウザ）が適切なモーダル性内のレンダリングと、ファウンデーション・クラス中にハード・コーディングされたモーダル性の間の適切な同期化とを扱えるようにする。一例は、すべてのファウンデーション・クラスがJava Classとして提供される場合であろう。これにより、対話モデルによるプログラミングをJavaアプレットまたはサーブレットに拡張することが可能である。

【０１９４】（iv）対話によるハイブリッド・プログラミングは、宣言型と命令型の結合である。すなわち、ＣＦＣおよびＣＦＣを使用して（かつ、よりタスク特定に）構築された他のオブジェクト、例えばjavaアプレットへの呼出しを伴うＣＭＬページである。したがって、対話プログラミング・モデルによるプログラミングは、一般にすべてのプログラミング・モデルをカバーすると考えられる。

【０１９５】（ｖ）スクリプティング。ＣＭＬは、再利用したいと思うどんなスクリプティング（http://www.ecma.ch/stand/ecma-262.htmに定義されているＥＣＭＡスクリプトなど）もマルチモーダル・シェルのスクリプティング言語として直接サポートすることができる。モーダル性特定のスクリプト（JavascriptやＷＭＬスクリプトのような）は、モーダル性特定のスクリプティング言語として考えなければならない。ＣＭＬのＥＣＭＡスクリプトがどのようにレガシー・ブラウザ用に変換されることになるかのより詳細な挙動を今日（すなわち我々が今日のインフラストラクチャを使用する場合のステップのために）定義することは可能だが、これらは、モーダル性特定として（すなわち画像のように）単純に扱うことができる。

【０１９６】まとめとして、本発明の構成に関して以下の事項を開示する。

【０１９７】（１）ユーザから１つまたは複数のコンピュータ・ベースの装置を介してアクセス可能なアプリケーションをプログラムする方法であって、前記アプリケーションにアクセスするために使用される前記１つまたは複数のコンピュータ・ベースの装置とユーザが対話ベースのプログラミング・コンポーネントによって行うことが可能な対話を表現するステップを含み、前記対話ベースのプログラミング・コンポーネントが、前記アプリケーションに関連するコンテンツ／アプリケーション論理およびプレゼンテーション要件に対して独立、さらに、コンポーネントごとにトランスコーディングされ

て、前記１つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの１つまたは複数のモーダル性特定のレンダリングを生成する方法。
（２）前記アプリケーションの少なくとも一部がサーバから、クライアントの役割を果たす前記１つまたは複数のコンピュータ・ベースの装置のうちの少なくとも１つにダウンロードされるクライアント／サーバ構成において、前記サーバに常駐する前記コンテンツ／アプリケーション論理への接続を提供するように動作可能なコードを前記アプリケーション中に含めるステップをさらに含む、上記（１）に記載の方法。
（３）前記コンテンツ／アプリケーション論理接続コードが、前記アプリケーションに関連する１つまたは複数のデータ・モデル、属性制約、および妥当性検査規則のうちの少なくとも１つを表す、上記（２）に記載の方法。
（４）前記１つまたは複数のモーダル性特定のレンダリングが、前記アプリケーションの一部のスピーチ・ベースの表現を含む、上記（１）に記載の方法。
（５）前記スピーチ・ベースの表現がVoiceＸＭＬに基づく、上記（４）に記載の方法。
（６）前記１つまたは複数のモーダル性特定のレンダリングが、前記アプリケーションの一部の視覚ベースの表現を含む、上記（１）に記載の方法。
（７）前記視覚ベースの表現がＨＴＭＬ、ＣＨＴＭＬ、ＷＭＬのうちの少なくとも１つに基づく、上記（６）に記載の方法。
（８）前記ユーザ対話が前記対話ベースのプログラミング・コンポーネントによって宣言的に表現される、上記（１）に記載の方法。
（９）前記ユーザ対話が前記対話ベースのプログラミング・コンポーネントによって命令的に表現される、上記（１）に記載の方法。
（１０）前記ユーザ対話が前記対話ベースのプログラミング・コンポーネントによって宣言的かつ命令的に表現される、上記（１）に記載の方法。
（１１）前記対話ベースのプログラミング・コンポーネントが、前記ユーザと前記１つまたは複数のコンピュータ・ベースの装置との間で発生する可能性のあるダイアログに関連する基本要素を含む、上記（１）に記載の方法。
（１２）前記対話ベースのプログラミング・コンポーネントが複合要素を含み、前記複合要素は、前記ユーザと前記１つまたは複数のコンピュータ・ベースの装置との間で発生する可能性のあるダイアログに関連する２つ以上の前記基本要素の集合体である、上記（１１）に記載の方法。
（１３）前記対話ベースのプログラミング・コンポーネントの１つが会話型ジェスチャを表す、上記（１）に記載の方法。
（１４）前記会話型ジェスチャが、ユーザへの情報メッセージをカプセル化するジェスチャを含む、上記（１３）に記載の方法。
（１５）前記会話型ジェスチャが、コンテキスト・ヘルプ情報をカプセル化するジェスチャを含む、上記（１３）に記載の方法。
（１６）前記会話型ジェスチャが、別のジェスチャの完了が成功したときに行われるアクションをカプセル化するジェスチャを含む、上記（１３）に記載の方法。
（１７）前記会話型ジェスチャが、イエス／ノー・ベースの質問をカプセル化するジェスチャを含む、上記（１３）に記載の方法。
（１８）前記会話型ジェスチャが、ユーザが選択肢のセットから選択することを期待される場合のダイアログをカプセル化するジェスチャを含む、上記（１３）に記載の方法。
（１９）前記選択ジェスチャが前記選択肢のセットを表すサブ要素を含む、上記（１８）に記載の方法。
（２０）前記選択ジェスチャが、選択がパスすべきテストを表すサブ要素を含む、上記（１８）に記載の方法。
（２１）前記選択ジェスチャが、前記テストが不合格の場合に呈示すべきエラー・メッセージを表すサブ要素を含む、上記（２０）に記載の方法。
（２２）前記会話型ジェスチャが、所与の会話型ジェスチャの結果を妥当性検査するための規則をカプセル化するジェスチャを含む、上記（１３）に記載の方法。
（２３）前記会話型ジェスチャが、文法処理規則をカプセル化するジェスチャを含む、上記（１３）に記載の方法。
（２４）前記会話型ジェスチャが、ユーザが前記アプリケーションの各部分をナビゲートするのを助けるダイアログをカプセル化するジェスチャを含む、上記（１３）に記載の方法。
（２５）前記会話型ジェスチャが、少なくとも１つのユーザ・ログインおよび認証の情報を求める要求をカプセル化するジェスチャを含む、上記（１３）に記載の方法。
（２６）前記会話型ジェスチャが、制約付きのユーザ入力を求める要求をカプセル化するジェスチャを含む、上記（１３）に記載の方法。
（２７）前記会話型ジェスチャが、制約のないユーザ入力を求める要求をカプセル化するジェスチャを含む、上記（１３）に記載の方法。
（２８）前記会話型ジェスチャが、情報のサブミットを制御するジェスチャを含む、上記（１３）に記載の方法。
（２９）論理入力イベント、ならびに、前記論理入力イベントと定義された前記論理入力イベントをトリガする物理入力イベントとの間の関連を定義する機構を提供するステップをさらに含む、上記（１）に記載の方法。

（３０）前記コンポーネントごとのトランスコーディングがＸＳＬ変換規則に従って行われる、上記（１）に記載の方法。
（３１）前記コンポーネントごとのトランスコーディングがJava Beanに従って行われる、上記（１）に記載の方法。
（３２）前記コンポーネントごとのトランスコーディングがJava Server Pageに従って行われる、上記（１）に記載の方法。
（３３）前記対話ベースのプログラミング・コンポーネントによるプレゼンテーションが、前記１つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの前記１つまたは複数のモーダル性特定のレンダリングを同期させることを可能にする、上記（１）に記載の方法。
（３４）前記対話ベースのプログラミング・コンポーネントによるプレゼンテーションが自然言語理解環境をサポートする、上記（１）に記載の方法。
（３５）前記１つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの前記１つまたは複数のモーダル性特定のレンダリングに関連するプレゼンテーション・フィーチャを表面的に変更することを可能にするコードを含めるステップをさらに含む、上記（１）に記載の方法。
（３６）コンポーネントごとにトランスコーディングして前記１つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの前記１つまたは複数のモーダル性特定のレンダリングを生成するための規則を変更することを可能にするコードを含めるステップをさらに含む、上記（１）に記載の方法。
（３７）実装されている、基礎をなすデータ・モデルの定義が、前記ユーザ対話を定義するマークアップ言語から分離される、上記（１）に記載の方法。
（３８）node_id属性が各コンポーネントに付加され、前記属性は様々な出力の上にマッピングされる、上記（１）に記載の方法。
（３９）モーダル性特定のマークアップ・コンポーネントをカプセル化する通過機構が作成者に提供される、上記（１）に記載の方法。
（４０）前記コンポーネントが並列に活動化させることができる、上記（１）に記載の方法。
（４１）前記プレゼンテーションおよびトランスコーディングが拡張可能である、上記（１）に記載の方法。
（４２）前記アプリケーションの状態がカプセル化される、上記（１）に記載の方法。
（４３）前記表現が動的に生成されるデータの参照を可能にし、前記コンテンツ／アプリケーション論理へのコールバック機構をサポートする、上記（１）に記載の方法。
（４４）１つまたは複数のコンピュータ・ベースの装置に関連するアプリケーションにアクセスする際に使用する装置であって、１つまたは複数のプロセッサを備え、前記プロセッサが、（ｉ）アプリケーション・サーバから前記アプリケーションを得るように動作可能であり、前記アプリケーションが、前記１つまたは複数のコンピュータ・ベースの装置とユーザが対話ベースのプログラミング・コンポーネントによって行うことが可能な対話によってプログラム的に表現され、前記対話ベースのプログラミング・コンポーネントが、前記アプリケーションに関連するコンテンツ／アプリケーション論理およびプレゼンテーション要件に対して独立し、前記プロセッサはまた、（ii）前記対話ベースのプログラミング・コンポーネントをコンポーネントごとにトランスコーディングして、前記１つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの１つまたは複数のモーダル性特定のレンダリングを生成するように動作可能である装置。
（４５）前記１つまたは複数のプロセッサが前記１つまたは複数のコンピュータ・ベースの装置にわたって分散される、上記（４４）に記載の装置。
（４６）前記アプリケーションの少なくとも一部がサーバから、クライアントの役割を果たす前記１つまたは複数のコンピュータ・ベースの装置のうちの少なくとも１つにダウンロードされるクライアント／サーバ構成において、前記サーバに常駐する前記コンテンツ／アプリケーション論理への接続を提供するように動作可能なコードを前記アプリケーション中に含めるステップをさらに含む、上記（４４）に記載の装置。
（４７）前記コンテンツ／アプリケーション論理接続コードが、前記アプリケーションに関連する１つまたは複数のデータ・モデル、属性制約、および妥当性検査規則のうちの少なくとも１つを表す、上記（４６）に記載の装置。
（４８）前記１つまたは複数のモーダル性特定のレンダリングが、前記アプリケーションの一部のスピーチ・ベースの表現を含む、上記（４４）に記載の装置。
（４９）前記スピーチ・ベースの表現がVoiceＸＭＬに基づく、上記（４８）に記載の装置。
（５０）前記１つまたは複数のモーダル性特定のレンダリングが、前記アプリケーションの一部の視覚ベースの表現を含む、上記（４４）に記載の装置。
（５１）前記視覚ベースの表現がＨＴＭＬ、ＣＨＴＭＬ、ＷＭＬのうちの少なくとも１つに基づく、上記（５０）に記載の装置。
（５２）前記ユーザ対話が前記対話ベースのプログラミング・コンポーネントによって宣言的に表現される、上記（４４）に記載の装置。
（５３）前記ユーザ対話が前記対話ベースのプログラミング・コンポーネントによって命令的に表現される、上記（４４）に記載の装置。

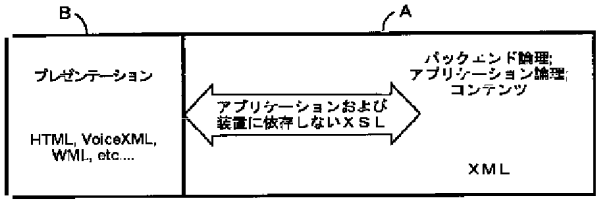
（５４）前記ユーザ対話が前記対話ベースのプログラミング・コンポーネントによって宣言的かつ命令的に表現される、上記（４４）に記載の装置。
（５５）前記対話ベースのプログラミング・コンポーネントが、前記ユーザと前記１つまたは複数のコンピュータ・ベースの装置との間で発生する可能性のあるダイアログに関連する基本要素を含む、上記（４４）に記載の装置。
（５６）前記対話ベースのプログラミング・コンポーネントが複合要素を含み、前記複合要素は、前記ユーザと前記１つまたは複数のコンピュータ・ベースの装置との間で発生する可能性のあるダイアログに関連する２つ以上の前記基本要素の集合体である、上記（５５）に記載の装置。
（５７）前記対話ベースのプログラミング・コンポーネントの１つが会話型ジェスチャを表す、上記（４４）に記載の装置。
（５８）前記会話型ジェスチャが、ユーザへの情報メッセージをカプセル化するジェスチャを含む、上記（５７）に記載の装置。
（５９）前記会話型ジェスチャが、コンテキスト・ヘルプ情報をカプセル化するジェスチャを含む、上記（５７）に記載の装置。
（６０）前記会話型ジェスチャが、別のジェスチャの完了が成功したときに行われるアクションをカプセル化するジェスチャを含む、上記（５７）に記載の装置。
（６１）前記会話型ジェスチャが、イエス／ノー・ベースの質問をカプセル化するジェスチャを含む、上記（５７）に記載の装置。
（６２）前記会話型ジェスチャが、ユーザが選択肢のセットから選択することを期待される場合のダイアログをカプセル化するジェスチャを含む、上記（５７）に記載の装置。
（６３）前記選択ジェスチャが前記選択肢のセットを表すサブ要素を含む、上記（６２）に記載の装置。
（６４）前記選択ジェスチャが、選択がパスすべきテストを表すサブ要素を含む、上記（６２）に記載の装置。
（６５）前記選択ジェスチャが、前記テストが不合格の場合に呈示すべきエラー・メッセージを表すサブ要素を含む、上記（６４）に記載の装置。
（６６）前記会話型ジェスチャが、所与の会話型ジェスチャの結果を妥当性検査するための規則をカプセル化するジェスチャを含む、上記（５７）に記載の装置。
（６７）前記会話型ジェスチャが、文法処理規則をカプセル化するジェスチャを含む、上記（５７）に記載の装置。
（６８）前記会話型ジェスチャが、ユーザが前記アプリケーションの各部分をナビゲートするのを助けるダイアログをカプセル化するジェスチャを含む、上記（５７）に記載の装置。
（６９）前記会話型ジェスチャが、少なくとも１つのユーザ・ログインおよび認証の情報を求める要求をカプセル化するジェスチャを含む、上記（５７）に記載の装置。
（７０）前記会話型ジェスチャが、制約付きのユーザ入力を求める要求をカプセル化するジェスチャを含む、上記（５７）に記載の装置。
（７１）前記会話型ジェスチャが、制約のないユーザ入力を求める要求をカプセル化するジェスチャを含む、上記（５７）に記載の装置。
（７２）前記会話型ジェスチャが、情報のサブミットを制御するジェスチャを含む、上記（５７）に記載の装置。
（７３）論理入力イベント、ならびに、前記論理入力イベントと定義された前記論理入力イベントをトリガする物理入力イベントとの間の関連を定義するための機構を提供するステップをさらに含む、上記（４４）に記載の装置。
（７４）前記コンポーネントごとのトランスコーディングがＸＳＬ変換規則に従って行われる、上記（４４）に記載の装置。
（７５）前記コンポーネントごとのトランスコーディングがJava Beanに従って行われる、上記（４４）に記載の装置。
（７６）前記コンポーネントごとのトランスコーディングがJava Server Pageに従って行われる、上記（４４）に記載の装置。
（７７）前記対話ベースのプログラミング・コンポーネントによるプレゼンテーションが、前記１つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの前記１つまたは複数のモーダル性特定のレンダリングを同期させることを可能にする、上記（４４）に記載の装置。
（７８）前記対話ベースのプログラミング・コンポーネントによるプレゼンテーションが自然言語理解環境をサポートする、上記（４４）に記載の装置。
（７９）前記１つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの前記１つまたは複数のモーダル性特定のレンダリングに関連するプレゼンテーション・フィーチャを表面的に変更することを可能にするコードを含めるステップをさらに含む、上記（４４）に記載の装置。
（８０）コンポーネントごとにトランスコーディングして前記１つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの前記１つまたは複数のモーダル性特定のレンダリングを生成するための規則を変更することを可能にするコードを含めるステップをさらに含む、上記（４４）に記載の装置。
（８１）実装されている、基礎をなすデータ・モデルの定義が、前記ユーザ対話を定義するマークアップ言語か

ら分離される、上記（４４）に記載の装置。
（８２）node_id属性が各コンポーネントに付加され、前記属性は様々な出力の上にマッピングされる、上記（４４）に記載の装置。
（８３）モーダル性特定のマークアップ・コンポーネントをカプセル化する通過機構が作成者に提供される、上記（４４）に記載の装置。
（８４）前記コンポーネントが並列に活動化させることができる、上記（４４）に記載の装置。
（８５）前記プレゼンテーションおよびトランスコーディングが拡張可能である、上記（４４）に記載の装置。
（８６）前記アプリケーションの状態がカプセル化される、上記（４４）に記載の装置。
（８７）前記表現が動的に生成されるデータの参照を可能にし、前記コンテンツ／アプリケーション論理へのコールバック機構をサポートする、上記（４４）に記載の装置。
（８８）前記１つまたは複数のプロセッサが前記１つまたは複数のコンピュータ・ベースの装置にわたって分散され、前記アプリケーションが前記１つまたは複数のコンピュータ・ベースの装置にまたがって同期化される、上記（４４）に記載の装置。
（８９）前記アプリケーションの表現がさらに、１つまたは複数のモーダル性特定のマークアップ言語を介して前記１つまたは複数のモーダル性特定のレンダリングを表面的に変更することを可能にする、上記（４４）に記載の装置。
（９０）ユーザからの１つまたは複数のコンピュータ・ベースの装置を介したアプリケーションへのアクセスを提供する際に使用するブラウザ装置であって、コンピュータ実行可能なコードを含む機械読取可能媒体を備え、前記コンピュータ実行可能なコードが、実行時に、アプリケーション・サーバからアプリケーションを得るステップであって、前記アプリケーションが、前記１つまたは複数のコンピュータ・ベースの装置とユーザが対話ベースのプログラミング・コンポーネントによって行うことが可能な対話によってプログラム的に表現され、前記対話ベースのプログラミング・コンポーネントが、前記アプリケーションに関連するコンテンツ／アプリケーション論理およびプレゼンテーション要件に対して独立しているステップと、前記対話ベースのプログラミング・コンポーネントをコンポーネントごとにトランスコーディングして、前記１つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの１つまたは複数のモーダル性特定のレンダリングを生成するステップとを実施することを可能にするブラウザ装置。（９１）ユーザから１つまたは複数のコンピュータ・ベースの装置を介してアクセス可能なアプリケーションをプログラミングする際に使用する製造品であって、コンピュータ実行可能なコードを含む機械読取可能媒体を備え、前記コンピュータ実行可能なコードが、実行時に、前記アプリケーションにアクセスするために使用される前記１つまたは複数のコンピュータ・ベースの装置とユーザが対話ベースのプログラミング・コンポーネントによって行うことが可能な対話を表現するステップを実施することを可能にし、前記対話ベースのプログラミング・コンポーネントが、前記アプリケーションに関連するコンテンツ／アプリケーション論理およびプレゼンテーション要件に対して独立、さらに、コンポーネントごとにトランスコーディングされて、前記１つまたは複数のコンピュータ・ベースの装置上で前記アプリケーションの１つまたは複数のモーダル性特定のレンダリングを生成する製造品。

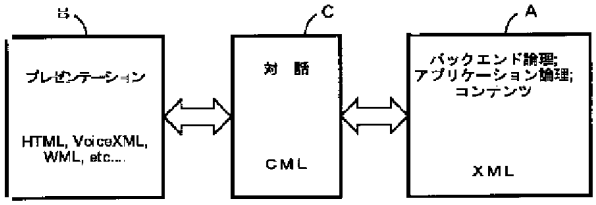
【図面の簡単な説明】
【図１】従来のアプリケーション・プログラミング手法を示す図である。
【図２】本発明の対話ベースのアプリケーション・プログラミング手法を示す図である。
【図３】本発明の一実施形態によるＣＭＬオーサリングされたアプリケーションの一例を示す図である。
【図４】ＸＦＯＲＭＳ概念を示す図である。
【図５】本発明の対話ベースのプログラミング手法におけるＸＦＯＲＭＳの使用を示す図である。
【図６】本発明の対話ベースのプログラミング手法におけるＸＦＯＲＭＳの使用を示す図である。
【図７】ＣＭＬソース・コード・ページから変換されたＧＵＩ歓迎ページをＨＴＭＬブラウザで見たものを示す図である。
【図８】ＣＭＬソース・コード・ページから変換されたＧＵＩ歓迎ページをＨＴＭＬブラウザで見たものを示す図である。
【図９】ＣＭＬソース・コード・ページから変換されたＧＵＩ歓迎ページをＨＴＭＬブラウザで見たものを示す図である。
【図１０】ＣＭＬソース・コード・ページから変換されたＧＵＩ歓迎ページをＷＭＬブラウザでみたものを示す図である。
【図１１】ＨＴＭＬによって表面変更されたＣＭＬソース・コード・ページから変換されたＧＵＩ歓迎ページをＨＴＭＬブラウザで見たものを示す図である。
【図１２】ＭＶＣモデルの新しい解釈を示す図である。
【図１３】既存のシステムから本発明によってＣＭＬを完全に使用するまでの移行ロード・マップを示す図である。
【図１４】既存のシステムから本発明によってＣＭＬを完全に使用するまでの移行ロード・マップを示す図である。
【図１５】既存のシステムから本発明によってＣＭＬを完全に使用するまでの移行ロード・マップを示す図である。

【図１６】本発明の一実施形態によるマルチモーダル・ブラウザ体系を示す図である。
【図１７】本発明のマルチモーダル・ブラウザ機構の一実施形態によるアプリケーション・プログラミング過程におけるＣＭＬの例示的な使用を示す流れ図である。
【図１８】本発明のマルチモーダル・ブラウザ機構の一実施形態によるアプリケーション・プログラミング過程におけるＣＭＬの例示的な使用を示す別の流れ図である。
【図１９】本発明の一実施形態による複数装置ブラウザ体系を示す図である。
【符号の説明】
１０　ＣＭＬコード
２０　ジェスチャ：タイトル
２２　ジェスチャ：メッセージ
２４　ジェスチャ：排他的選択
６０　マルチモーダル・ブラウザ
６２　マルチモーダル・シェル
６４　視覚ブラウザ
６４　ＧＵＩレンダリング・ブラウザ
６６　スピーチ・ブラウザ
６６　スピーチ・レンダリング・ブラウザ
６８　会話型エンジン
６９　コンテンツ・サーバ
７６　登録テーブル
７７　モーダル性特定のレンダリング
７８　モーダル性特定のレンダリング
８０　サーバ
８２　クライアント装置
８４　サーバ
８６　オーディオ入力／出力機能
９４　ＭＭシェル登録テーブル
９８　ジェスチャＸＳＬ規則

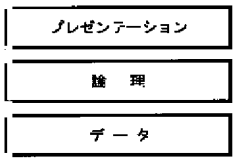
【図１】



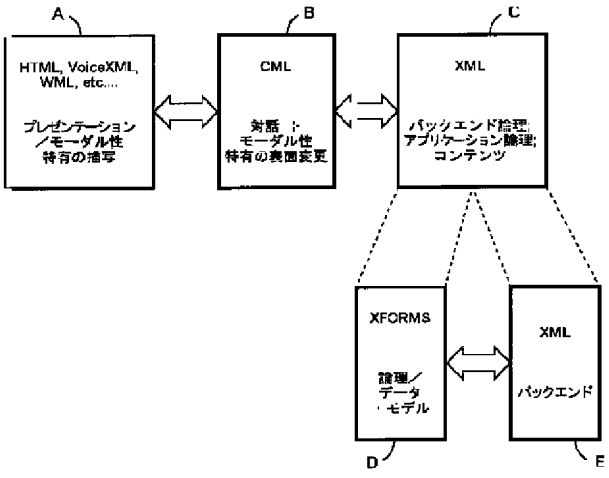
【図２】



【図４】

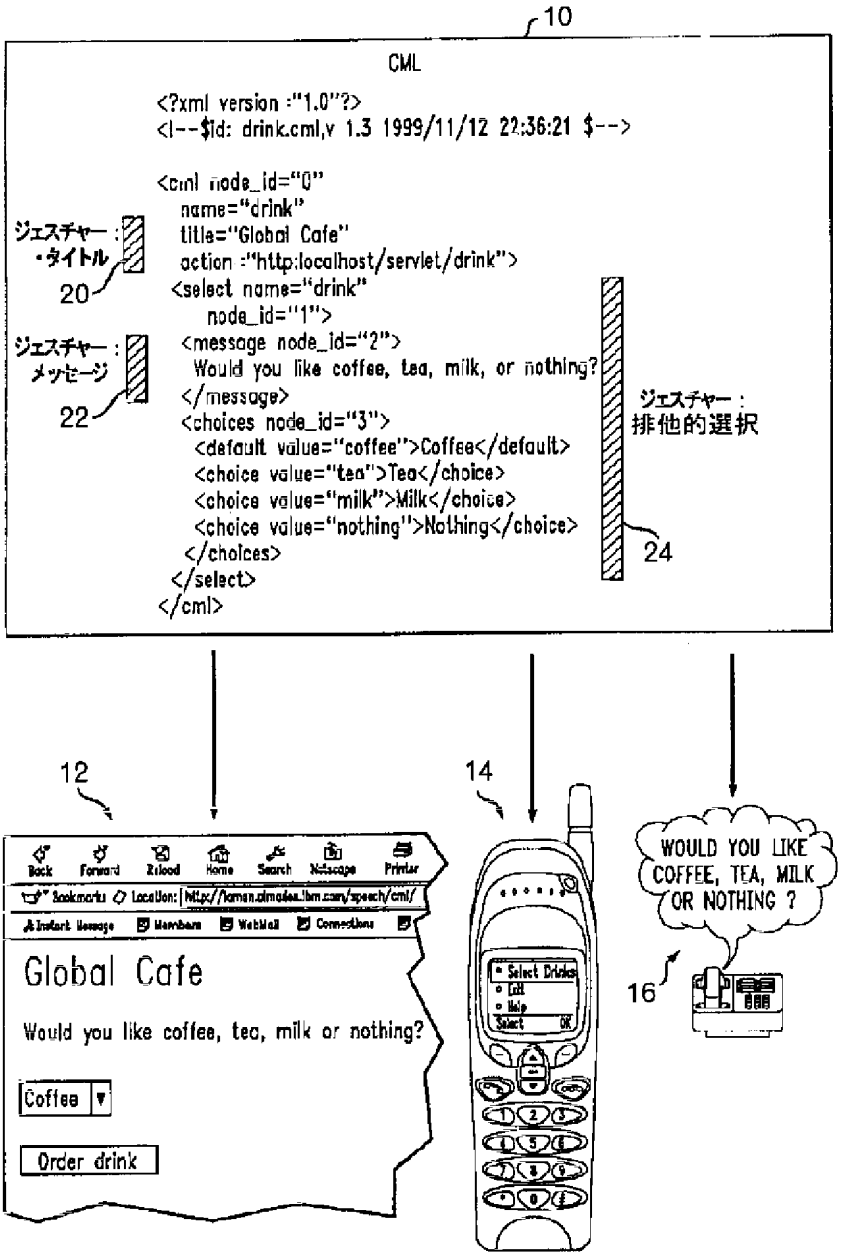


【図５】

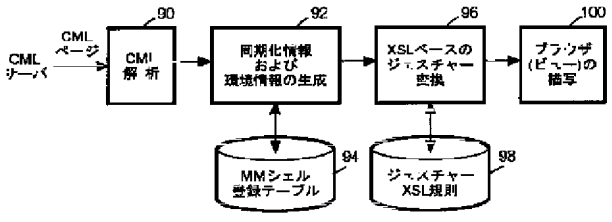


【図１０】

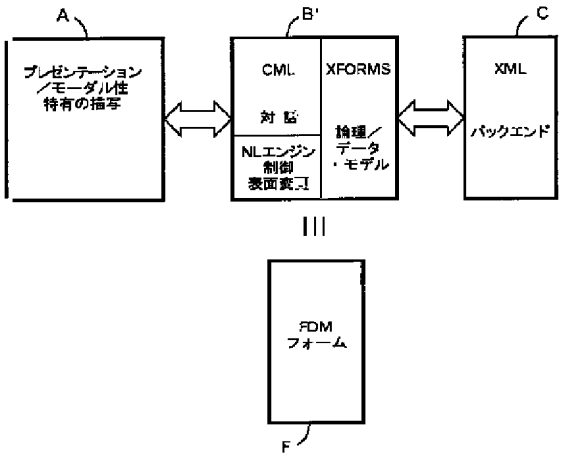
【図3】


10
CML
<?xml version :"1.0"?>
<!--$Id: drink.cml,v 1.3 1999/11/12 22:36:21 $-->
<cml node_id="0"
name="drink"
title="Global Cafe"
action :"http:localhost/servlet/drink">
<select name="drink"
node_id="1">
<message node_id="2">
Would you like coffee, tea, milk, or nothing?
</message>
<choices node_id="3">
<default value="coffee">Coffee</default>
<choice value="tea">Tea</choice>
<choice value="milk">Milk</choice>
<choice value="nothing">Nothing</choice>
</choices>
</select>
</cml>
ジェスチャー：
・タイトル
20
ジェスチャー：
メッセージ
22
ジェスチャー：
排他的選択
24
12
14
16
Global Cafe
Would you like coffee, tea, milk or nothing?
Coffee
Order drink
WOULD YOU LIKE COFFEE, TEA, MILK OR NOTHING ?

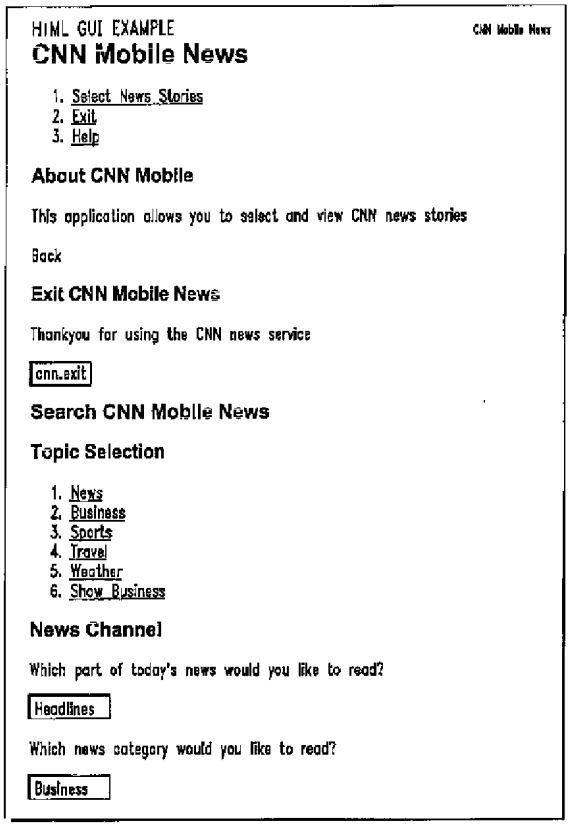
【図18】


CML
サーバ
CML
ページ
90
CML
解析
92
同期化情報
および
環境情報の生成
96
XSLベースの
ジェスチャー
変換
100
ブラウザ
(ビュー)の
描写
94
MMシェル
登録テーブル
98
ジェスチャー
XSL規則

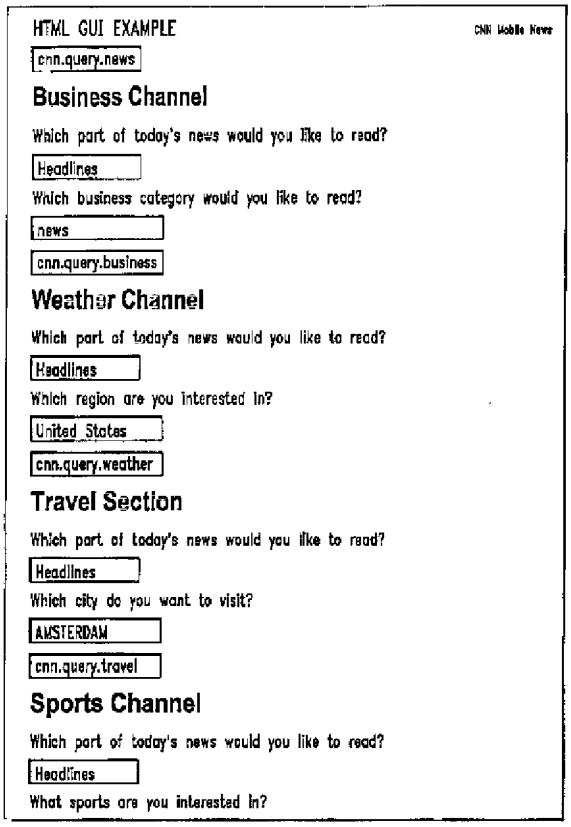
【図6】



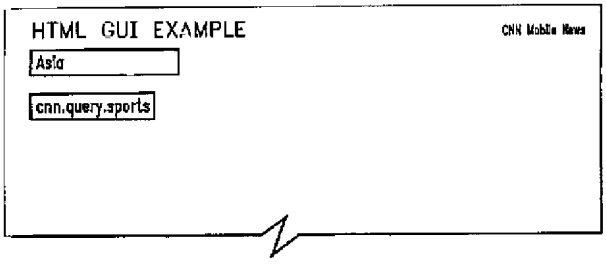
【図7】



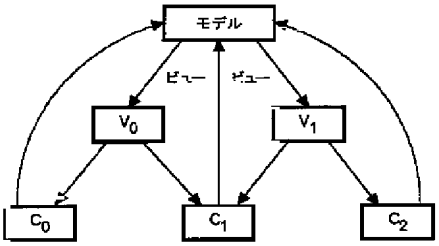
【図8】



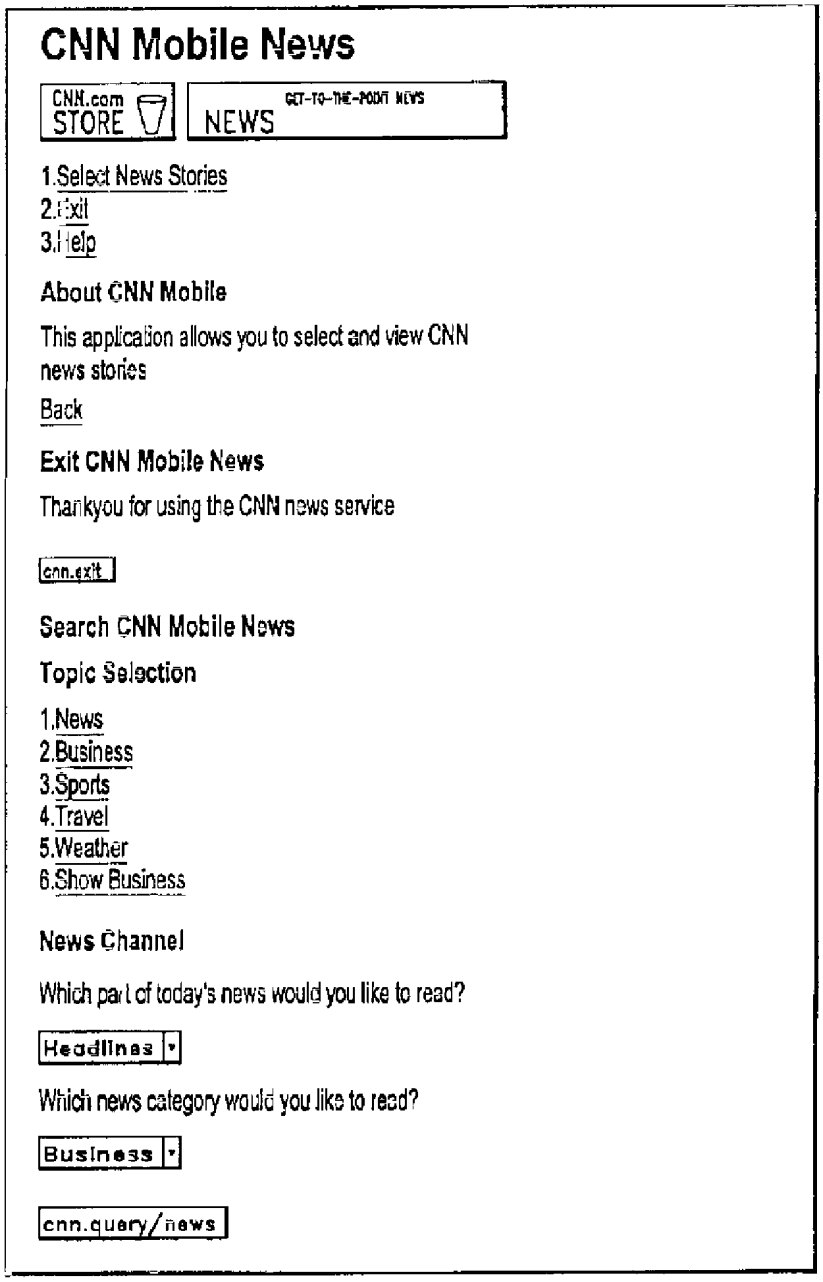
【図9】



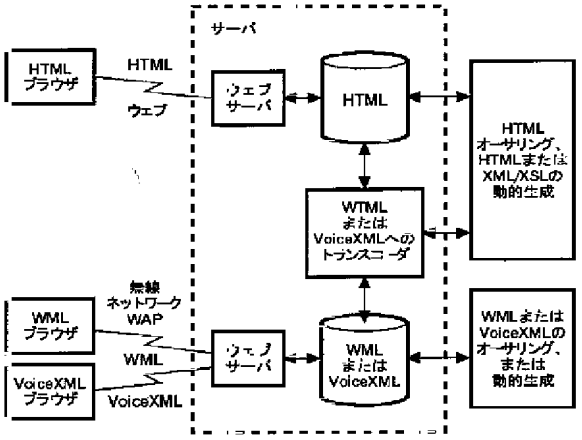
【図12】



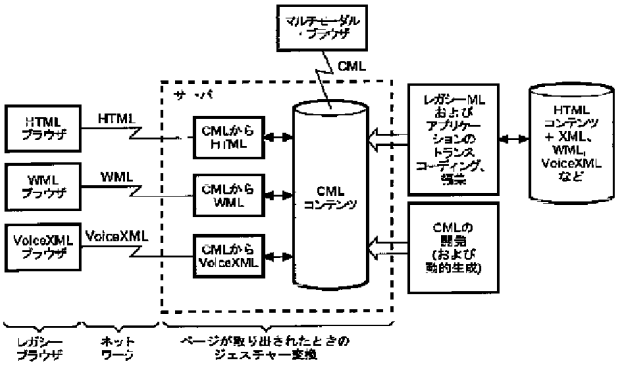
【図11】


CNN Mobile News
CNN.com STORE
GET-TO-THE-POINT NEWS
NEWS
1.Select News Stories
2.Exit
3.Help
About CNN Mobile
This application allows you to select and view CNN news stories
Back
Exit CNN Mobile News
Thankyou for using the CNN news service
cnn.exit
Search CNN Mobile News
Topic Selection
1.News
2.Business
3.Sports
4.Travel
5.Weather
6.Show Business
News Channel
Which part of today's news would you like to read?
Headlines
Which news category would you like to read?
Business
cnn.query/news

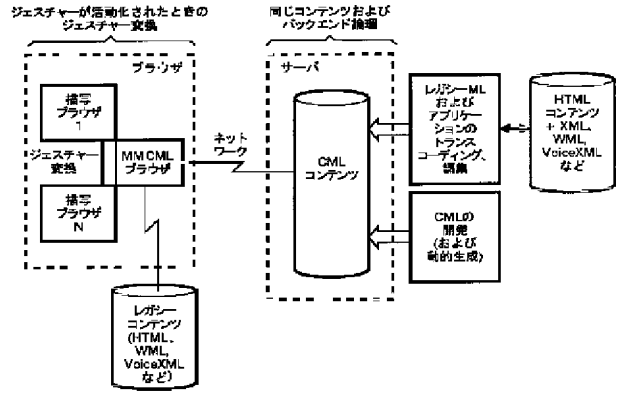
【図１３】



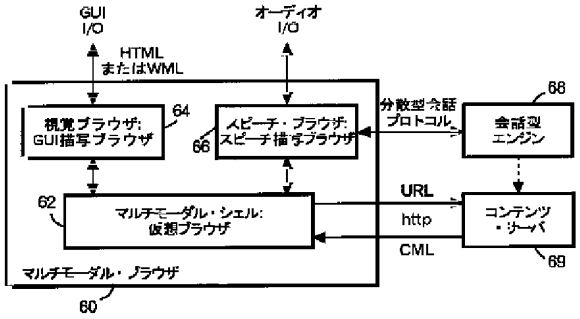
【図１４】



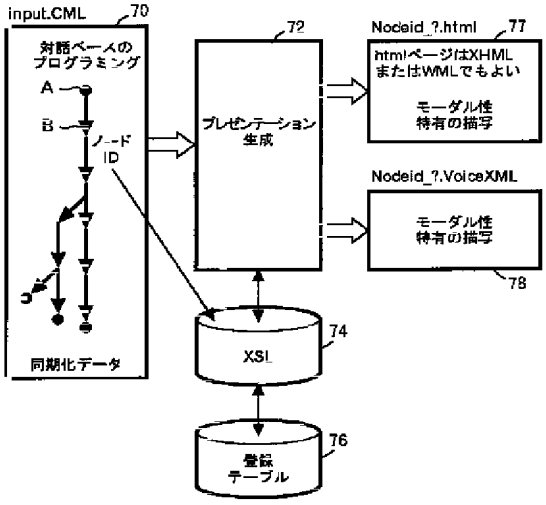
【図１５】



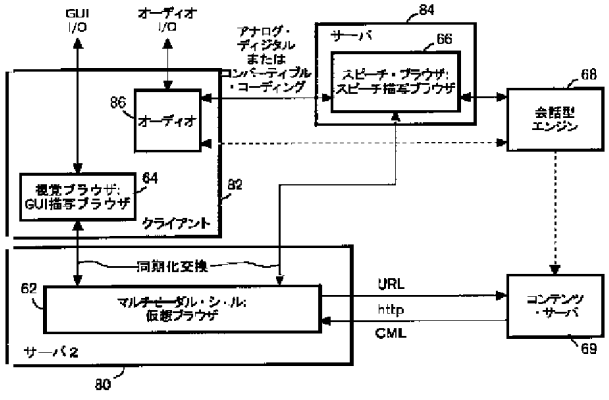
【図１６】



【図１７】



【図１９】



フロントページの続き

(72)発明者　ステファン・ハーマン・マエス
アメリカ合衆国06811　コネチカット州ダンベリー　ウィンターグリーン・ヒル・ロード　1

(72)発明者　ティルヴィルヴァマ・ライヴィー・ラマン
アメリカ合衆国95123－5310　カリフォルニア州サンノゼ　ラ・テラス・サークル　2715